Making the Most of Your Data: KDD Cup 2007 "How Many Ratings" Winner's Report

Saharon Rosset*
IBM T.J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598
srosset@us.ibm.com

Claudia Perlich
IBM T.J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598
perlich@us.ibm.com

Yan Liu
IBM T.J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598
liuya@us.ibm.com

ABSTRACT

We describe the ideas and methodologies that we developed in addressing the KDD Cup 2007 *How Many Ratings* task, and discuss how they contributed to our success.

1. DRIVERS OF MODELING SUCCESS

At the Data Analytics Research group at IBM Research we aim to combine theoretical rigor with practical usefulness in our research and the projects we develop for IBM groups and external customers. Our projects often include aspects of data analysis, algorithm development, application development and product delivery [4; 3; 1]. Based on our experience there are several important components to success in modeling data — whether it be in a competition or in real-life modeling problems. One possible characterization of these components divides them into three general categories:

1. **Data and domain understanding.** The focus here is on understanding how the data and modeling problem were generated; how the data can best be used to address the problem at hand; what transformations or preprocessing are required to make the data most appropriate; and how this knowledge can be put together in a coherent manner.

2. **Statistical insights.** This aspect of the modeling process concentrates on using our data and domain understanding to generate models and insights that are statistically sound and optimal. The statistical aspect is distinct from the data understanding one in relying on probabilistic and statistical insights rather than knowledge about the data generating processes.

3. **Modeling or learning approach.** This is the step that typically generates the most "scientific" interest in the data mining and machine learning communities, of developing and/or choosing the best algorithmic approach to solve the modeling problem at hand.

From our experience, the ordering of these categories above is consistent with their typical importance in maximizing the success of practical modeling projects. The ability to understand the data and the domain well, figure out their correct interpretation and appropriate use is by far the most useful way to gain "an edge" and improve models above and beyond what any statistical insights or modeling approaches can. Correctly formulating a statistical or probabilistic framework is also of critical importance, when possible. Finally, in our opinion, the learning approach, while highly influential on bottom-line performance in many cases, cannot be counted on as a way to circumvent the need to understand the data and the statistical setup properly.

In this short paper we use this classification of the elements of modeling success to describe our approach. After describing the general setup of the challenge in Section 2, we detail the data insights we used in Section 3. We discuss out key statistical insights in Section 4, and show how everything comes together to guide our modeling approach in Section 5. Finally, we briefly analyze the competition results and how the different components of our approach affected our performance in Section 6.

2. PROBLEM SETUP

The second task in the KKD-CUP was to predict the total number of reviews that a movie received during 2006 from the universe of users in the Netflix competition training set. This task can be viewed as a regression problem where the number of ratings that a movies receives in a given period of time depends on a number of factors that contribute to the popularity and in turn to the number of ratings of a movie. Such factors include age, arrival in the Netflix database, genre, rating and also the characteristics and history of the roughly 480,000 reviewers.

These factors naturally do not only impact the number of ratings in the current time frame but also the number of rating in previous periods. This suggests a temporal dynamic in the rating with different periods in the movie life-cycle: prior to Box-office release, prior to DVD release, prior to availability in Netflix and finally the slow decrease of interest as the movie ages. So the historical reviewing behavior of a movie is another vital piece of information to capture the dynamic life-cycle of a movie. Such lagged rating counts can be extracted from the Netflix competition dataset with time-stamped ratings from 1998 through 2005.

One way of formalizing the supervised modeling problem of the *How Many Ratings* task is to build a time-series model that estimates the number of rating in one period as a function of past ratings and movie-specific features on historical

*Current address: Raymond and Beverly Sackler School of Mathematical Sciences, Tel Aviv University, Israel

data and roll the model over to the next time period.

However, there is another less obvious approach that can be taken to formalize The *How Many Ratings* task as a supervised learning problem that takes advantage of the *Who reviewed what* test set as discussed in the next section.

3. DOMAIN OBSERVATIONS

We will discuss two important observations about the generation of the test set for the KDD-CUP and training data that strongly affect the design of our modeling approach.

3.1 Using *Who Reviewed What* test set to model *How Many Ratings*

The two tasks for the KDD-CUP were constructed based on the 2006 reviews of the 17770 movies in the Netflix competition dataset. The organizers randomly assigned 8863 movies to the *How Many Ratings* task and the remaining movies were used to construct the test set for *Who Reviewed What*. Let us take a more detailed look at the construction of this test set and how it can be utilized to build a model for the *How Many Ratings* task.

In order to achieve a reasonably high base rate for the classification task *Who Reviewed What*, the sampling probability for a movie,user pair was based on the product of the marginal rating distributions by movie and user in 2006. The marginal is proportional to the number of ratings a movie received in 2006. So the number of movie appearnaces in a test pair is proportional to the total number of reviews the movie received in 2006. This suggests an interesting modeling approach. We can use the appearance in the *Who Reviewed What* test set as the dependent variable to estimate a model to predict the number of 2006 ratings. We can then apply this model to the movies in the *How Many Ratings* test set. This idea has two major advantages:

1. we capture the dynamics of the 2006 ratings; and
2. we can make optimal use of all recent data up to end of 2005 to construct independent variables

However, there are two issues to consider. We are still missing a scaling parameter. The counts in the test set of *Who Reviewed What* are relative to a sample of 100,000 movie-rating pairs. What remains unknown and of critical importance is the total number of reviews in 2006 to use as a scaling factor for our prediction. This modeling problem is described in detail in Section 5.

Another important observation is the fact, that the counts in the *Who Reviewed What* test set are not really proportional to the marginal because the organizers had to remove pairs that had received ratings prior to 2006. The probability of rejection is a function of the marginal distribution and affects popular movies more than others. We resolve this problem by correcting the counts as outlined in Section 4.2.

3.2 Dynamics of the total rating counts

One of the missing pieces of information is the scaling parameter that is needed to predict the total number of ratings in 2006. To appropriately address this point, we had to understand the dynamics of how the total number of ratings that all movies changes over time — both calendar time and time in the *life-cycle* of the movies being reviewed. We noticed some interesting discrepancies in the behavior of the total number of reviews over time, in particular a steep drop in the number of reviews in the fourth quarter of 2005, which we were only partially able to explain through movie and reviewer life cycles. Since this did not prove a major influence on our model's performance, we defer detailed discussion to a longer version of this paper.

4. STATISTICAL OBSERVATIONS

There are two statistical aspects to this data modeling problem that captured our attention.

4.1 Is Poisson the right likelihood?

Consider a set of m objects (in this case, movies) with counts $n_1, ...n_m$ (in this case, number of reviews per movie in a given period of length t). Our first observation is that under mild and reasonable assumptions about the arrival process of new reviews for each movie, these counts have a marginal Poisson distribution:

$$m_i \sim \text{Pois}(\lambda_i \cdot t)$$

Consequently, if we decide to use a linear model (or a kernel-based non-linear model) to describe the dependence of the observed movie counts on a set of features $\mathbf{x}_1, ..., \mathbf{x}_p$ (these are vectors of length n), a good candidate modeling approach would be a Poisson regression, a generalized linear model [2], with the natural (log) link function:

$$\begin{aligned} log(\lambda_i) &= \sum_j \beta_j x_{ij} \\ \hat{\beta} &= \arg\min_\beta \sum_i [\lambda_i \cdot t - n_i \cdot \log(\lambda_i \cdot t)] \end{aligned} \quad (1)$$

A more interesting situation is when we use the set of counts $\tilde{n}_1, ..., \tilde{n}_m$ from *Who Reviewed What* test set as our modeling target. This test set was sampled proportional to the true counts $n_1, ...n_m$ (subject to the rejection sampling correction we discuss next), and is constrained to sum to a fixed number (say, 100000). It is easy to show that:

$$\tilde{n}_1, ..., \tilde{n}_m | \sum_i \tilde{n}_i = 100000 \sim \text{Multinomial}(100000, p_1, ..., p_m)$$

where $p_i = \lambda_i / \sum_k \lambda_k$ is the relative rate of movie i.

Now, if we look at each of the $\tilde{n}_i$'s, their "marginal" conditional distribution is Binomial$(100000, p_i)$ and since this is a *large n, small p* situation, the distribution of $\tilde{n}_i$ is well approximated by Pois$(100000, p_i)$ distribution. Further, it can be shown that although the binomial are not independent (the dependence is created by the constraint on their sum), this does not invalidate a Poisson regression approach for maximum likelihood estimation of their parameters, since Poisson regression guarantees that the sum of the predictions will equal the sum of the observations [2]. Thus we propose to use a similar formulation to Eq. (1):

$$\hat{\beta} = \arg\min_\beta \sum_i [\lambda_i - \tilde{n}_i \cdot \log(\lambda_i)]$$

where we have eliminated the known time period t, and we will have to estimate a *scaling factor* as discussed in the previous section to scale the estimate λ_i's to use them for prediction.

4.2 Rejection sampling correction

Our discussion in the previous section assumed that the $\tilde{n}_i$'s were sampled *proportionally* from the original n_i's. As we discussed in the previous section, this is not exactly true, because after this proportional sampling, some of the sampled movies were rejected, based on previously having been ranked (prior to 2006). To obtain $\tilde{n}_i$s that are indeed proportionally sampled this rejection would have to be inverted. Here we describe our algorithm for this inversion.

Let $p_i = \lambda_i / \sum_k \lambda_k$ be the *true* sampling rate for movie i, and $q_j = \eta_j / \sum_l \eta_l$ be the corresponding sampling rate for user j. We estimate p and q as follows Suppose the sample size 100,000 is large enough for us to estimate p_i and q_j. We have a hidden variable, that is, the number of samples rejected because they have appeared before 2006, which we denoted by N. We observe n_i appearances of movie i in the final sample set, which satisfies:

$$E[n_i|N] = p_i(100,000 + N)(1 - \sum_{t \in U_t} q_t), \qquad (2)$$

where U_t is the set of users that has reviewed the movie t before 2006. On the right hand side of eq(2), the first product corresponds to the total number of samples with movie i (before rejection), and the last term is the proportion of pairs that are been eliminated because they appear before 2006. Similarly, we observe m_j appearances of user j in the final sample set, which satisfies:

$$E[m_j|N] = q_j(100,000 + N)(1 - \sum_{k \in M_k} p_k), \qquad (3)$$

where M_k is the set of movies that has been reviewed by user j before 2006. We implemented an ad-hoc iterative procedure for solving the equations (2,3), by alternating between fixing the q_j's and solving (2), and fixing the p_k's and solving (3). This gives us a more accurate estimate of p_k, q_j and N (our interest is, of course focused on the p_k's). This correction can be thought of as increasing the marginal for movies that are likely to have been rejected a lot, because they have been heavily reviewed before 2006, while also taking into account which reviewers reviewed them.

5. MODELING APPROACH

The culmination of all the discussion in the previous sections led us to a modeling approach that is outlined graphically in Figure 1:

1. Extract a list of features for each movie:

- log(Number of reviews by month for the most recent quarter+1)
- log(Number of reviews by quarter for the most recent year+1)
- log(Number of reviews by year for the last four years+1)
- Movie's age in the Netflix database (days since first review), capped at two years, and also transformed into log and square scale.
- Some characteristics of the movie's ratings (% of 5's, average rating, etc.)
- Movie's Genre (taking only the most common genres and binning all the rest into "other")

2. Use the test set of *Who Reviewed What* as a *response* for training a model:

- Apply the sampling correction discussed in Section 4.2.
- Build a Poisson regression model describing log(Poisson rate for movie i in *Who Reviewed What* 2006 test set) as a function of the features extracted from the full Netflix dataset (i.e., all reviews until 12/2005, including those in the Netflix qualifying set)

3. Go through a separate modeling exercise to estimate the *scaling factor*, i.e., the total number of reviews that were given to all movies in 2006:

- Create four *lagged* datasets, which are "anchored" in previous quarters. For example, a lagged dataset for Q205 would only contain movies and reviewers which appeared in the Netflix data before end of June 2005. This is consistent with our modeling approach in some of our business modeling projects [4].
- Build four predictive models which use subsequent quarters as response. For the Q205 lagged dataset, we build a model which uses Q405 review numbers as response, *when limited to the set of movies and reviewers who were active by end of Q205.* This gives us a *two quarters ahead* prediction model.
- These models can now be applied to our complete dataset to predict numbers of reviews in 2006. For example, applying the model built on the Q205 lagged dataset with Q405 as response, to the full Netflix dataset would comprise a prediction for Q206 (two quarters ahead of Q405).
- This prediction can be used either as an actual prediction for 2006 movie review counts, if it is better than the predictions generated by the models we built in step 2 above (see below on evaluation strategies for determining whether this is the case); or, it can be used to determine the scaling factor between the *Who Reviewed What* test set and the total review numbers. This scaling factor can then be applied to the model's predictions on the *How Many Ratings* movies.

This schematic description glosses over many details, like feature selection, interaction selection, exact form of the Poisson regression models, etc. We next discuss in some detail the elements of model evaluation and model selection.

5.1 Evaluation, validation and model selection

Our best asset for evaluation is the same as for modeling — the *Who Reviewed What* test set, after the rejection sampling correction. It can be used in a straight forward manner to evaluate the models built on it, through a cross validation approach or training-test splits.

For models built on the lagged datasets, the exercise is less trivial. To use *Who Reviewed What* test set for evaluation we need to invert the sampling scheme. To avoid various complications that stem from this, and to give the lagged models the best opportunity to surpass the *Who Reviewed What*-based models in terms of performance, we actually find the best possible scaling parameter to the lagged models predictions in terms of their performance on *Who Reviewed What* test set.

The end result of all these evaluations are model-performance scores for all models we consider for prediction. The best performances in terms of log-scale on holdout data from *Who Reviewed What* for two model classes — *Who Reviewed What*-based vs. lagged data-based — were 0.24 and 0.31, respectively. We concluded that we should use the models we

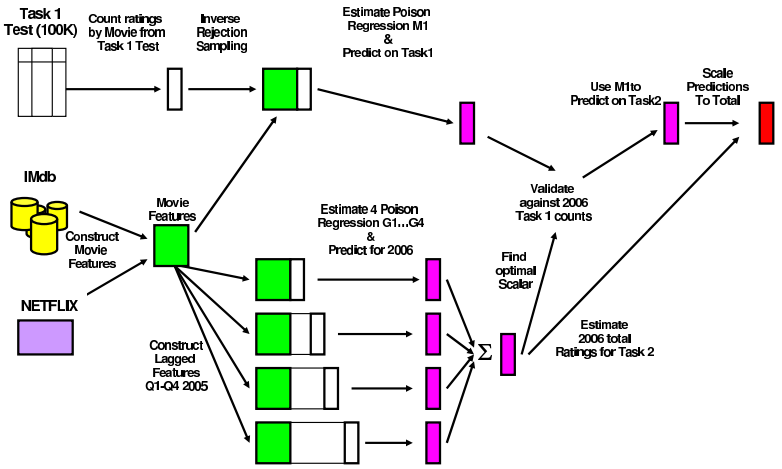

Figure 1: A schematic of our overall modeling approach

build on *Who Reviewed What* test set for prediction, and the models built on lagged datasets for scaling only.

6. ANALYSIS OF COMPETITION RESULTS

The log-scale MSE of our winning model on the *How Many Ratings* task 2006 reviews was 0.263 as shown in Table 1. This error has two components: The error of the model for the scaled-down *Who Reviewed What* test set (which we estimated at about 0.24); and the error from our incorrect scaling factor, i.e., the mismatch between the scaling factor we estimated from the lagged models and the true correct scaling factor.

In our case, the sum of our predictions was 9.35 million, and the sum of true responses was 8.7 million. Table 2 details the scores we would have attained if we had scaled our predictions differently. By correctly scaling to 8.7 million total, we would have attained MSE of about 0.234. Our lowest possible MSE could have been 0.208 with a scaling factor of 0.8. This is possibly due to the behavior of Poisson noise under the log transformation: the roughly symmetric noise (for large Poisson parameter) becomes long-left-tailed under the log transformation, and hence consistent under-prediction may lead to better performance.

7. CONCLUSION

We can summarize our KDD Cup 2007 *How Many Ratings* experience in three short bullets:

- **We had fun** dealing with the data and understanding it, trying out different modeling approaches and speculating about outcomes.
- **We did well** and we believe that a combination of reasons drove this success, but possibly our "bootstrapping" of *Who Reviewed What* test set for training was the most important factor.
- **We encountered some interesting research problems**, most notably the *inverse rejection sampling* problem discussed in Section 4.2. While we applied the inversion here to correct an artifact of the competition sampling, we expect that this inversion problem may be encountered in various real-life problems.

Table 1: Results for the top performing teams on the *How Many Ratings* task

Team	Score (RMSE)	MSE
IBM Research	0.513	0.263
NeoMetrics	0.523	0.273
Inductis	0.541	0.292

Acknowledgments

We thank Rick Lawrence and Zhenzhen Kou for help in useful discussions and in data formatting.

8. REFERENCES

[1] R. Lawrence, C. Perlich, S. Rosset, et al. Analytics-driven solutions for customer targeting and sales force allocation. *IBM Systems Journal*, 46(4), 2007.

[2] P. McCullagh and J. Nelder. *Generalized Linear Models*. Chapman and Hall, London, 1989.

[3] C. Perlich, S. Rosset, R. Lawrence, and B. Zadrozny. High quantile moldeing for customer wallet estimation with other applications. In *KDD07*, 2007.

[4] S. Rosset and R. Lawrence. Data enhanced predictive moldeing for sales targeting. In *SIAM Data Mining*, 2006.

Table 2: The effect of scaling on competition MSE. The first column is a hypothetical scaling factor applied to our submitted predictions, the second is the implied total for 2006, and the third the competition score.

Scaling	Total (mil.)	MSE	Comment
0.7	6.5	0.222	
0.8	7.5	**0.208**	Best performance
0.9	8.4	0.225	
0.93	**8.7**	0.234	Correct scale
1	9.35	**0.263**	Our actual score
1.1	10.285	0.316	