My solution to the PAKDD 2014 -ASUS Malfunctional Components Prediction Competition

Shize Su
Email: ss5vq@virginia.edu
PhD Student at University of Virginia, U.S.A

1. Summary

In my opinion, this competition is more like a curve fitting and extrapolation problem rather than a usual machine learning competition problem. Therefore, usual machine learning modeling techniques were not used here.

The key analysis task has been placed in analyzing the pattern of repair rate as a function of time for each module-component and then to construct a reasonable repair rate pattern for prediction period. I use R (mainly for data preprocessing and prediction output) together with Excel (mainly for data pattern visualization and analysis) for this competition.

My final model is quite simple; for each component, I use last 8 months monthly repair data (May 2009-Dec 2009) only, and use a piece-wise exponentially decay function for prediction (p.s.: inspired by the PAKDD competition forum discussion "Sample-submission Benchmark in Leader board", thanks), with the exponentially decay function parameters tuned by observation on the pattern of these 8 months repair data as well as the leaderboard feedback.

2. Features Selection/Extraction

First, I completely ignore the sales data. Although I agree that there exist some correlation between these sales data and the repair rate in the prediction period, I think it might be hard and very time consuming to find a good way to utilize these sales data to improve prediction performance. I made this decision also partially due to the fact that when I made my 2nd submission in this competition only using such 8 months repair rate data with a simple piece-wise exponentially decay model, it already gave me Top 10 standing position at that time (Scored about 2.6). So I decide to not to spend time to deal with the sales data but focus on improving my current model with the repair rate data.

Second, for repair rate pattern analysis, I use R and excel to generate the monthly repair data (for missing data, put 0 there) in May 2009-Dec 2009 for each module-component combination in the test set, and sort the data based on the sum of these 8 months

repair data. The reason why I initially try to only use last 8 months repair data is : 1) since sales stop at Feb 2008, my intuition is that the repair rate pattern near (or before) Feb 2008 would be quite different from (and thus less predictive and less useful) the repair rate pattern in prediction period (Jan 2010-Jul 2011), and the repair rate pattern in time period closer to prediction period should be more predictive for the repair rate pattern in prediction period; 2) inspired by the PAKDD competition forum discussion "Sample-submission Benchmark in Leader board" (suggest 5-10) . And this turns out work fairly well. In fact, when I make 2nd submission in this competition using only these 8 months repair data with a simple model, it already gives me Top 10 standing position.

Analysis on these last 8 months repair rate data allows observing some interesting facts, such as:

- Although there are 187 module-components in the test set, many of them have very few repairs (<10 in total for 8 months), while the 15 module-components (which I called "key module-components") with highest repairs contribute to about 90% of the total repairs of all 187 module components (based on May 2009-Dec 2009 repair data). This implies that better prediction accuracy of the repair rate of these "key module-component" might significantly improve the overall performance, while we could not improve overall performance too much by finding a better model than a simple exponential decay function for those module-components with few repairs. Therefore, it might be better to focus on improving prediction accuracy on these "key module-component".
- The repair rate pattern could vary a lot across different module-components for the same time period. This implies that it might be better to fit a specific function to each module-component, rather than fit a general function to all the module-component, at least for those "key module-components". Besides, based on my submission leader board feedback, I find that a piece-wise exponential decay function might perform better than a simple smooth exponential decay function.
- In the last 4 months (Sep 2009-Dec 2009), almost all module-components (except very few module-components which exhibit somewhat steady behavior) exhibit decaying behavior approximately. This implies that now it might be safe to assume that all module-components have passed the fluctuation period and enter a "consistent decay period". Thus, it might be safe to use decay function for prediction of the repair rate in Jan 2010-Jul 2011.

In short, in my final model, I only use the monthly repair data in May 2009-Dec 2012 as features for each module-component in the test set.

3. Modeling Techniques

I benefit a lot from the PAKDD forum discussions about all different method trials and ideas.

Initially inspired by the idea posted by Chitrasen under forum discussion "Sample submission Benchmark in Leader Board", I started with a simple piece-wise exponential decay function only using the monthly repair data in May 2009-Dec 2009, which gave me Top 10 standing position in my 2nd submission of this competition. This made me confident that this approach should work well for this data set.

Forum discussion about "How to go about cross validation" made me give up the trials for new methods like glm, gbm, nnet, etc. Instead, after conducting analysis and with the key observations as stated in part 2 (Feature Selection/Extraction), I decide to spend most of my efforts in improving the prediction accuracy of those "key module-components" by tuning the parameters of the piecewise exponential decay functions.

The fact that most module-components seems to enter consistent decay period in Sep 2009-Dec 2009, together with the 50%/50% public/private data split makes me believe that it would be feasible to get valuable information by fitting the leader board without risking over fitting too much. Therefore, the parameters in these exponential decaying functions are tuned both on observations about the repair rate pattern in May 2009-Dec 2009 as well as the leaderboard feedback. For example, M2P16 and M7P13 exhibit really different repair rate patterns in May 2009-Dec 2009 (See Table 1), and thus quite different parameters are used in the decaying function model (detailed parameter see R-code) for these two module-components.

Table 1 M2P16 and M7P13 Repair Data, May 2009-Dec 2009

Repair	May-09	Jun-09	Jul-09	Aug-09	Sep-09	Oct-09	Nov-09	Dec-09
M2P16	1583	1923	2501	2247	1753	834	580	367
M7P13	58	78	100	77	96	84	104	88

Besides, to better utilize the limited number of valuable submission shots, I also try to use one submission to fit several time spot values. In particular, for example, for M2P09, I set the repair rate for Apr 2011, May 2011, Jun 2011 and Jul 2011 to be 10000, 20000, 40000, and 80000 respectively, and I set all other repair rate for all module-components to be 0. Then I compare the public board score (62.00799) with 0 benchmark score (5.65179). Since the public board is calculated at 50% of the data, we can compute (62.00799-5.65179)*4256*50% =119926 which is approximately 120000, and this indicates that only Jun 2011(repair 40000 in submission) and Jul 2011(repair 80000 in

submission) is in public board (this is true when 10000>>actual value of M2P09 repair rate in Apr 2011-Jul 2011, which I think would be a safe assumption based on training data analysis), and the sum of the actual values of repair rate in Jun 2011 and Jul 2011 is equal to (120000-119926)/2=37. Then I use this feedback to further refine the parameters in my piece-wise exponential decay function. I try one or two of such submission for those "key module-components" which I think a submission might be most probable to improve my overall model performance. For those module-components with high repair rate but I don't have a chance to try a submission, I then use its May 2009-Dec 2009 repair data pattern analysis together with other key module-components' tuned parameters analysis to come up with a reasonable guess of the appropriate parameter values.

Finally, I make some additional adjustments to further improve my model performance: 1) for one module-component, if the total repair amount in May 2009-Dec 2009 is less or equal to 5, then I just predict 0 for all months in prediction period; 2) if the piece-wise exponential decay function predicted monthly repair<0.3, then I predict 0 instead; 3) if the piece-wise exponential decay function predicted monthly repair<0.7 and time is later than Nov 2010, I also predict 0 instead.

4. Code Description

Please see documentation in the code.

In short, there are two parts: 1) the first part is data preprocessing using R together with Excel; 2) the second part is about predictions using R.

In detail, in first part, I generate the monthly repair rate in May 2009-Dec 2009 for each module-component in the test set, and sort the data using the sum of these 8 months monthly repair rate. The result of this step is the "trainForExcelAnalysis.csv" file.

In the second part, I implemented piece-wise exponential decay function to output the prediction of monthly repair rate in Jan 2010-Jul 2011 for each module-component in test set. The result of this step is the "predsubmit.csv" file, which can be directly submitted to Kaggle without further processing.

5. Dependencies

Software R 3.0.2 and Excel 2010 is used in the code and data processing.

6. How to Generate the Solution

Please look at the README file.

It consists of an R script as well as a .csv file “trainForExcelAnalysis.csv” .

Input data files for the R script are “RepairTrain.csv” and “Output_Target_Mapping.csv” as provided on Kaggle competition website. These files should be located in R default working directory.

Running first part of the R script will generate the “trainForExcelAnalysis.csv” file, which is monthly repair data in May 2009-Dec 2009 for each module-component in test set. This file is used for repair pattern visualization and analysis within Excel (especially for analyzing “**key module-components**” repair rate pattern) to design appropriate parameters of piece-wise exponential decay function in the second part of R script (together with leaderboard feedback).

The second part of the R script will generate a “predsubmit.csv” file in the same R working directory, which can be submitted to Kaggle without any further processing.

7. Additional Comments

Here are some observations which are true for this competition but not necessarily true for other competitions:

- Excel seems to be an appropriate tool for repair rate data processing (e.g., sort), visualization and pattern analysis in this competition. For example, if you have ever once looked at these 8 monthly repair data (sorted) in May 2009-Dec2009, you will have a quite clear idea about which module-components’ repair rate prediction can be safely zero out. You can also get a clear idea about how the repair rate patterns vary among different “key module-components”.
- Since the sale stopped at Feb 2008, the repair rate pattern near 2008 (or before 2008) would be quite different from the repair rate pattern in the prediction period (Jan 2010-Jul 2011), and only the repair rate pattern in time period closest to prediction period would be most predictive of the repair rate pattern in prediction period. Also, the fact that most module-components seems to enter consistent decay period in Sep 2009-Dec 2009, implies that we can get valuable information about repair rate pattern from public board data fitting without risking over fitting much.

8. References

Wikipedia. Exponential decay. http://en.wikipedia.org/wiki/Exponential_decay