# Maintenance Action Recommendation Using Collaborative Filtering

Santanu Das[1]

[1] *NASA Ames Research Center, Moffett Field, CA, 94035*
*Santanu.Das-1@nasa.gov*

**ABSTRACT**

The problem we were trying to solve in 2013 PHM Society Conference Data Challenge competition [1] is closely related to remote monitoring and diagnostics in industrial applications. This data was generated from an industrial piece of equipment with a sensor network to measure several parameters and an onboard condition monitoring system. The measured data goes through a control logic in order to monitor the equipment's operating regime. At any time instant when some of these parameters meet a specific condition, the control system generates an unique event id/code. Each case is described by a set of event codes which characterize the atypical operating condition of the equipment. Some of these cases with specific event code combinations may be operationally significant and could be indicative of "Problem Types", some of which are assumed to be known to the subject matter experts. As a response to these problems, domain experts recommend appropriate diagnostic measures (or maintenance actions) depending on the problem types. The goal of this data competition is to build an automated system that can recommend particular maintenance action(s) to mitigate these problem(s).

## 1. RECOMMENDER SYSTEMS

Though active research has been conducted in recommender systems for the last two decades, more recently this field has gained tremendous attention particularly in service industry. In simple terms the intention of building recommender systems is to effectively suggest users on possible items of interest with some prior understanding of the contextual information on that user's and/or similar user's and/or similar item's profile. The recommended item(s) is the one with highest estimated "utility metric" that best represents user's notion of interest. Such "utility metric" include user's rating expressed explicitly in terms of symbols (stars) or numbers (binary, categorical). In more formal terms, the objective is to maximize user's utility function ($F_u$) in the joint user-item space ($U \times C \rightarrow \Re$) defined by a set of $U$ users where any user $u \in U$ and a set of $C$ items/commodities where any item $c \in C$. The recommender system's goal is to propose a new item $c^* \in C$ to the user $u$, such that $F_u(c, u)$ is maximized over all $c \in C$.

In this paper we have demonstrated the application of collaborative filtering technique to recommend maintenance actions. However, first we will provide a comprehensive overview on some of the popular recommendation techniques. Based on the literature (Koren & Bell, 2011; Koren, Bell, & Volinsky, 2009; Adomavicius & Tuzhilin, 2005), recommender systems are broadly classified into three groups, namely content filtering, collaborative filtering and hybrid approach which is essentially a combination of collaborative and content filtering. Here filtering is synonymous to estimating $F_u(c, u)$ using either item's content based features e.g., keyword weights or user's explicit feedback on items e.g., ratings. In a loose sense, content filtering based approach uses low level information whereas collaborative filtering uses high level information to accomplish the job. Content filtering assumes access to item's content ($D_c$). In this approach the first step is content based information retrieval where a set of attributes/features ($A_c$) that best describes an item $c$ are extracted. This is followed by estimating $F_u(c, u)$ based on the extracted features of items. We will elaborate this with an example. In the aviation safety domain, suppose an aviation safety officer (end user) is interested in reading a particular class of text reports written by the flight crew/passengers, for example turbulence related reports, then our goal is to recommend the safety inspector with similar reports on turbulence related topic based on his/her preference. In order to do that, we first need to know what are the important keywords that best describes turbulence related topic, e.g., "mountain', "clear-air", "convective" etc. There are several approaches in retrieving such information (keywords) and the choice of the approach typically depends on the nature of the data. For example, if we intend to find the keywords from a corpus of (only) turbulence related documents, we may use "*Term Frequency-Inverse Document Frequency* (TFIDF)" to measure the importance of the words across of the whole corpus. We will provide further details on TFIDF in another section. However if the document corpus contains various topics (e.g.,



[1] http://www.phmsociety.org/events/conference/phm/13/challenge/

weather, maintenance, turbulence, mechanical failure etc.) in varying proportions, then a complex machinery like "Latent Dirichlet Allocation (LDA)" is required to obtain a set of keywords to describe each topic. We refer the reader to the paper (Blei, Ng, & Jordan, 2003) for further details as it is beyond the scope of this paper. Once we obtain the key features ($A_c$) then the remaining task is to adopt a heuristics or a model based approach to compute the score of the utility function $F_u(A_c,D_{c_j})$ over all new items' content $D_{c_j}$. The item with the highest score or the *n* top scoring items are recommended to the user. In the above example, the contents of new reports that closely match the description of turbulence related topic are presented to the safety officer.

As mentioned earlier collaborative filtering based approach does not assume access to item's content $D_c$, instead it expects user's explicit feedback (ratings $r_c$) on items i.e. user-item interactions ($Y_{\hat{u}\hat{c}}$). One can think of $Y_{\hat{u}\hat{c}}$ as a matrix whose rows represent the set of users $\hat{u} \subset U$ who have provided ratings on a subset of items $\hat{c} \subset C$ and the columns represent the set of items, each of which has been at least rated by one or more users. Since users generally provide feedback not necessarily on every items but a small subset of items, $Y_{\hat{u}\hat{c}}$ tends to be extremely sparse. In collaborative filtering a new item is recommended to a user $\hat{u}_k$ based on the ratings of other users $\hat{u}_j$ having similar taste. In heuristics based collaborative filtering we first retrieve neighboring users $u_j$ and $u_k$ and then use an aggregate function to compute an average rating for each item from $\hat{c}$. The item with maximum weight (in this case rating) is recommended to user $u_k$. The readers are advised to consult the following literatures (Koren & Bell, 2011; Koren et al., 2009; Adomavicius & Tuzhilin, 2005) for detailed description of variations on aggregation functions.

Model based collaborative filtering however relies on mapping the user-item interaction to a latent space to achieve this goal using techniques like matrix factorization (Hoyer, 2002; Wold, Esbensen, & Geladi, 1987; Hoyer, 2004; Lee & Seung, 2001; Berry, Browne, Langville, Pauca, & Plemmons, 2006; Su & Khoshgoftaar, 2009; Cichocki & Zdunek, 2007; Koren & Bell, 2011; Koren et al., 2009), probabilistic mixture model (Condliff, Lewis, & Madigan, 1999) etc. In this paper we will restrict our discussion to matrix factorization methods only. In general matrix factorization methods are easy to understand and interpret from a mathematical perspective and also flexible in many sense. We will elaborate on some of these advantages. In these techniques we represent the user-item interaction $Y_{\hat{u}\hat{c}}$ as a product of two entities, one representing a collection of mixing factors (basis vectors) and the other represents target profiles. In some occasions these latent factors can be interpretable. In the "blind source separation" literature a frequently cited example is the "cocktail party problem" where sounds (signals) from multiple sources are mixed up resulting in a combined sound (signal) and the problem to solve is to separate these unique sound (signal) sources. With the assumption that the observed sound is a linear combination of the signals from unique sound sources, our objective is to find a way to compute not only the mixing profiles (basis vectors/latent factors) but also the unique sound sources. In some other domain these latent factors can be completely unobserved and at the same time not interpretable. In general matrix factorization techniques maps the given user-item interaction $Y_{\hat{u}\hat{c}}$ into a low dimensional (*m*) latent factor space defined by the inner product of $W_{\hat{u}m}$ and $H_{m\hat{c}}$, where $W_{\hat{u}m}$ represents the item-latent factor relationship and $H_{m\hat{c}}$ infers to what extent an user is interested in any item when expressed in terms of that item's relationship with one or more latent factor(s).

There are a wide variety of literatures (Hoyer, 2002; Wold et al., 1987; Hoyer, 2004; Lee & Seung, 2001; Berry et al., 2006; Su & Khoshgoftaar, 2009; Cichocki & Zdunek, 2007) on matrix factorization. Some of the popular and widely used algorithms in matrix factorization include Spectral Value Decomposition (SVD), Spectral Decomposition Algorithm (SDA) (Srivastava & Buntine, 1995), Non-Negative Matrix Factorization (NMF), Independent Component Analysis (ICA) and Principal Components Analysis (PCA). Although the fundamental idea is same but these machineries consider varying mathematical properties. For example, classical PCA results in a factorization of the matrix into a set of *m* orthogonal basis vectors. In addition this decomposition technique allows the elements of $W_{\hat{u}m}$ and $H_{m\hat{c}}$ to be either positive or negative. On the other hand Non-Negative Matrix Factorization finds this decomposition using squared reconstruction error criterion while preserving the non-negativity property over the entire solution space. As we will later demonstrate, some of these properties can be very useful in understanding the underlying data generating process. Researchers typically choose their favorite algorithm based on their understanding about the domain and depending on their goal. However standard matrix factorization techniques must be applied judiciously considering the amount of data or availability of priori knowledge or confidence on the observed data or issue related to over fitting and bias. The research community have proposed several variations (Cichocki & Zdunek, 2007; Koren & Bell, 2011; Koren et al., 2009; Adomavicius & Tuzhilin, 2005) of the standard matrix factorization techniques to take care of the above mentioned issues. Below we provide an overall summary of the state-of-the-art technologies on recommender systems. Further details on each of these bullets can be found in (Koren & Bell, 2011; Koren et al., 2009; Adomavicius & Tuzhilin, 2005).

1. **Content filtering**[Content based information retrieval, Content feature based information filtering]
   (a) *Heuristics based*[Similarity on item's content based features: e.g., cosine]
   (b) *Model based*[on item's content based features: e.g.,

classification, clustering etc.]

2. **Collaborative filtering**Rating based information retrieval, Rating based information filtering]
   (a) *Heuristics based*[Aggregation/similarity on user's explicit feedback (item's ratings): e.g., average/weighted average, cosine etc.]
      i. Variations of aggregation/similarity functions
   (b) *Model based*[on User's explicit feedback (item's ratings): e.g., latent factor based approach]
      i. Introducing sparsity
      ii. Introducing bias
      iii. Incorporating user's implicit feedback
      iv. Incorporating confidence on user's feedback
      v. Any combination of the above
3. **Hybrid filtering**[2 + 1]

## 2. THE COMPETITION PROBLEM

The problem we are trying to solve in this data challenge competition is closely related to remote monitoring and diagnostics in industrial applications. We were provided with three data sets for training. This data was generated from an industrial piece of equipment and no additional information was provided due to proprietary reasons. The equipment has a sensor network to measure several parameters with an onboard condition monitoring system. The measured data goes through a control logic in order to monitor the equipment's operating regime. At any time instant when some of these parameters meet a specific condition onboard, the control system generates an unique event id/code. It is possible that the control system concurrently triggers multiple events codes if all the necessary conditions are met at the same time instant. To clarify it further: think of any particular case as a set of observations either created by an automated system or manually by an engineer. Each case is described by a set of event codes which characterize the atypical onboard operating condition of the equipment. Some of these cases with specific event code combinations may be operationally significant and could be indicative of "Problem Types", some of which are assumed to be known to the subject matter experts. As a response to these problems, domain experts may recommend appropriate diagnostic measures (or maintenance actions) depending on the problem types described by a set of the events which have been flagged as a result of atypical operating condition. The goal of this data competition is to build an automated system that can recommend particular maintenance action(s) to mitigate these problem(s).

As mentioned earlier some of these cases are indicative of known "Problem Types" to the subject matter experts. However there could be many examples where a set of cases were not instructive enough to be acted on and hence termed as "Nuisance Cases". These cases were mostly created by automated systems and were presented to an subject matter expert who determined that the symptom was not sufficient to notify the customer of the identified "Problem Types". The task of recommending a particular maintenance action, gets relatively easier when these cases have less redundancies in the event combinations. The two major sources of redundancies in this analysis are strong background information of "Nuisance Cases" and overlapping events between "Problem Types". Given the context, the goal is to accurately recommend confirmed "Problem Types" and avoid making any recommendations for historical "Nuisance Cases". In this competition the metric used to measure success was in terms of the number of useful outputs. In equation form, the performance of the recommender system ($M$) was assessed using the following score:

$$Score(M) = N_o - N_{mc} - N_n \tag{1}$$

where $N_o$, $N_{mc}$ and $N_{ne}$ denotes the total number of outputs, the number of incorrect outputs and the number of nuisance outputs respectively. The number of incorrect outputs is essentially the penalty due to misclassifications i.e. the number of outputs where the true labels and recommender system's inferred labels do not match. On the other hand the number of nuisance outputs introduces the penalty due to false positives i.e. where the recommender system infers any truly "Nuisance Cases" as one of the "Problem Types", other than "none". While evaluating the $Score$ (refer Eqn.1) the number of cases with confirmed problems and nuisance test cases are essentially balanced. For the training set, the ground truth is know and hence $N_{mc}$ and $N_{ne}$ are known. However for the test set, a total of $174$ cases were correctly labeled by the subject matter experts with the known "Problem Types" and the rest as "Nuisance Cases", although these label information were not available to the participants during the competition. For the test case, the total number of outputs are compared with the known "Problem Types" ($174$ cases) and a fixed random sample of $174$ "Nuisance Cases". Below we have provided a summary on the data sets provided for training and test purposes.

1. Train:
   (a) Case to Problem ($F^{tr}_{cp}$) : file contains the different problems associated with each case. (used for training purpose with known "Problem Types")
   (b) Nuisance Cases ($F^{tr}_{cn}$) : file contains a set of cases that were not instructive enough to be acted on. (used for training purpose with unknown "Problem Types")
   (c) Case to Events and Parameters ($F^{tr}_{ces}$) : file contains the mapping between cases and events and snapshots of the parameters corresponding to each event.
2. Test:

Input: $F_{cp}^{tr}$, $F_{cn}^{tr}$, $F_{ces}^{tr}$, $F_{cp}^{tst}$

Step A: Data preparation
Step B: Information retrieval feature selection
Step C: Building model
Step D: Test set evaluation
Output: Unknown Labels on Test set

Figure 1. In this figure we show the basic steps of our approach while accomplishing the goal to predict a good number of unknown "Problem Types" which are truly operational while ignoring the majority of the test cases which may just be nuisance cases.

(a) Case to Problem ($F_{cp}^{tst}$) : file contains the different problems associated with each case. (used for testing purpose with unknown "Problem Types")

## 3. SUGGESTED APPROACH

With explicit feedback (preferences) from subject matter experts we want to predict a good number of unknown "Problem Types" which are truly operational while ignoring the majority of the test cases which may just be nuisance cases. There can be several ways one can approach this problem. For example, one can pose this as a binary classification problem where the objective will be to separate the truly operational cases from the nuisance cases using the class labels provided by the subject matter experts, followed by a subcategorization of the identified "Problem Types". In our approach, we assumed that the values of the parameters corresponding to each event may not be very useful on the first hand as the event id/code itself summaries the parameters settings for a specific condition. We also hypothesized that most of the "Problem Types", whether they are correctly identified or not, has to be inferred based on their observed event distributions. Apart from that, we have adopted a more generic approach where the key idea is to estimate the likelihood that a subject matter expert will accept a recommended maintenance action by characterizing both "Problem Types" and "Cases" in the event space. This approach also opens up further possibilities of ranking all of the potential choices of maintenance actions according to their likelihood. However in this competition we have recommended the maintenance action corresponding to the highest likelihood to the subject matter expert.

### 3.1. Data Preparation

In figure 1 we have shown the steps to accomplish the objective described above. In the very first step we construct the subject matter expert's understanding about the "Problem Types" given the corresponding cases i.e. "Problem Types"-"Case" correspondences $Y_{pc}$ from $F_{cp}^{tr}$ and similarly the observed "Event"-"Case" correspondences to $Z_{ec}$ from $F_{ces}^{tr}$. As mentioned earlier, we have ignored the values of the parameters corresponding to each event while constructing $Z_{ec}$. $F_{cp}^{tr}$ has 164 rows and two columns, namely case number and corresponding "Problem Types". $F_{cp}^{tr}$ has 10925 rows with the nuisance case numbers . The file ($F_{ces}^{tr}$) consisting of case to events mapping and parameter information has 1316653 rows and 32 columns. The columns represent case number, it's corresponding event id/code and 30 measured sensor values or parameters, the details of which are unknown. Figure 2 shows the detailed steps to construct $Y_{pc}$ and $Z_{ec}$. The data preparation recovered 10459 unique train cases and their corresponding 13 unique "Problem Types" and 1242 unique event ids. This means $Y_{pc}$ and $Z_{ec}$ are of the size of $10459 \times 13$ and $1242 \times 10459$ respectively. The test set $F_{ces}^{tst}$ was processes following the same logic as described in figure 2 to construct the "Event"-"Case" scenario ($Z_{ec}^{tst}$), except that we considered the same set of unique events ids from training set and ignored the rest of the events to maintain consistency. The size of $Z_{ec}^{tst}$ is $1242 \times 9358$.

Input: $F_{cp}^{tr}$, $F_{cn}^{tr}$, $F_{ces}^{tr}$
Step 1: [logcp, logces, logcn] = LOAD $F_{cp}^{tr}$,$F_{ces}^{tr}$,$F_{cn}^{tr}$.
Step 2: [logcp, logces, logcn] = FILTER logcp, logces, logcn by MISSING case number ;
Step 3: logec = EXCLUDE parameters FROM logces;
Step 4: conCase = UNION logec, logcn;
Step 5: logs = JOIN logcp and conCases BY case number;
Step 6: grpd = GROUP logs BY case number;
Step 7: FOREACH grpd GENERATE logs;
Step 8: uniqueCase, uniqueEventId, uniqueProblem = UNIQUE logs.Case, logs.EventIds, logs.Problems;
Step 9: $M_{ec}$ := FOREACH uniqueCase COUNT logs.EventIds;
Step 10: $Z_{ec}$ := $M_{ec}(M_{ec} > 1)$=1
Step 11: $Y_{cp}$ := FOREACH uniqueCase GENERATE logs.Problems;
Output: $Y_{cp}$, $Z_{ec}$

Figure 2. In this figure we provide the pseudo code of the data preparation steps. We were provided with three data files for training: file containing the different problems associated with each case, a second file containing a set of cases that were not instructive enough to be acted on and finally a file containing the mapping between cases and events and snapshots of the parameters corresponding to each event. The test file contains the different problems associated with each case.

### 3.2. Information Retrieval Feature Selection

In the field of information retrieval, the similarity between two entities is often measured by treating each entity as a vector of their occurrences as observed and computing distance between them using some heuristics. In this context, we can think of the columns from $Z_{ec}$ matrix as vectors representing the occurrences of the corresponding events over all cases that has been observed. In text mining, documents are often treated as collections of statistically independent words with varying frequencies i.e. documents can be represented

as vectors of frequencies with various word observed over the entire corpus. However not all these words are equally informative similar to the event id/cases in our problem. In the text mining paradiagm, to successfully retrieve the most relevant document from the entire collection we need to consider building a query using informative words which best describe the document (or similar documents) and at the same time minimize the background noise (avoid redundant words). In text mining community, Inverse Document Frequency (IDF) (Papineni, 2001) is one of the most popular measures of a word's importance. IDF of a word is measured by taking the logarithmic ratio of counts of total documents to the counts of documents that contains the specific word. Words which appear very frequently across documents are given lower weight as compared to words that appear less frequently among documents. The intuition is that highly frequent words across documents are not a good discriminator. Following the same analogy we generate the IDF scores of each event from $Z_{ec}$ and ranked them based on their scores. In our training data we have examples of both nuisance cases and problem cases. These nuisance cases along with the problem cases should be examined to determine which event features are redundant and not useful for discriminating "Problem Types" from the rest. The selection of event(s) is based on how the model ($M$) performs on the training data itself and the metric to evaluate the performance is shown in Eqn. 1; i.e. our goal is to maximize $Score(M_{\hat{E}})$. This means we select a subset $\hat{e}$ of the total events set $e$ based on their IDF scores and use these events as features to build a model $M_{\hat{E}}$ such that score 1 is maximized. The redundant events are all set to zeros. We will revisit this discussion again in the result section.

### 3.3. Building Model

In this section we will start discussing our model building approach in a more generalized way. However as we proceed further we will simplify the model to fit our problem. This is intentionally done to ensure that the intuition of the mathematical formulation is still consistent with the discussions above. We will start with a matrix decomposition algorithm that can reduce the number of dimensions in the observed signal and here we assume that a set of stationary signals is mixed through a linear mixing matrix. The result of this mixing matrix is the observed signal. The linear model can be expressed as:

$$\mathbf{Y} = \mathbf{WH} + \mathbf{E} \tag{2}$$

In this formulation, $\mathbf{Y} \in R^{p \times c}$ is a matrix of observed data or signals,$\mathbf{W} \in R^{p \times e}$ is the matrix of mixing or basis vectors, $\mathbf{H} \in R^{e \times c}$ represents the hidden source and the $\mathbf{E} \in R^{p \times c}$ matrix represents the noise sources. Here we also assume that there are a total of $e$ stationary components of $\mathbf{H}$. We will need to solve for $\mathbf{W}$ and $\mathbf{H}$ given the $\mathbf{Y}$ matrix. Our objective will be to minimize the square Euclidean distance $\mathbf{D(Y\|W,H)} = \frac{1}{2}||\mathbf{Y} - \mathbf{WH}||^2$. From the nature of the observed data we know $\mathbf{Y}$ is definitely sparse in nature but $\mathbf{H}$ may not. We don't make any assumption about the sparsity of $\mathbf{W}$. Instead we introduce $L-2$ regularization terms. In this context it is worth mentioning that the choice of $L-2$ regularization is very instinctive as we intend to restrict large value components and hence reduce chances of over fitting. The modified cost function looks like,

$$\mathbf{D(Y\|W,H)}^{\lambda_1,\lambda_2} = \frac{1}{2}||\mathbf{Y} - \mathbf{WH}||^2 + \lambda_1|||\mathbf{W}|||^2 + \lambda_2|||\mathbf{H}|||^2 \tag{3}$$

where $\lambda_1$ and $\lambda_2$ are user defined regularization parameters. We are essentially minimizing a cost function which is a regularized least-square function. It should be noted that we will also make non-negativity assumptions on $\mathbf{W}$ and $\mathbf{H}$. The above cost function can be made more general by introducing additional constraints like sparsity, offsets, weighting functions etc.

The decomposition procedure starts by assuming a random starting point for $\mathbf{W}$, computing $\mathbf{H}$ and then re-computing $\mathbf{W}$ given the current estimate of $\mathbf{H}$. The updates of $\mathbf{W}$ and $\mathbf{H}$ happen in such a fashion that the cost function is $\mathbf{D(Y\|W,H)}^{\lambda_1,\lambda_2}$ is minimized.The update equations are based on a least-squares solution to the problem and are derived from the gradients of the cost function (Eqn. 3) with respect to $\mathbf{W}$ and $\mathbf{H}$. When $\lambda_1 > 0$ and/or $\lambda_2 > 0$, the update scheme takes their effect into account and the cost function gets compensated as a result of the regularized terms. Although there exists several update schemes for $\mathbf{W}$ and $\mathbf{H}$ (e.g., multiplicative update (Lee & Seung, 2001)), least-squares solutions in particular are extremely favorable to large scale problems. The pseudo code of the update scheme is shown in Fig. 3. The above algorithm is also known as Alternating Least Square (ALS) NMF (Cichocki & Zdunek, 2007; Koren et al., 2009) with the non-negativity constraint on $\mathbf{W}$ and $\mathbf{H}$. For many applications the non-negativity assumption preserves important properties of the observed data and can lead to superior results in some cases. This may also lead to better and more interpretable solutions (Hoyer, 2002; Wold et al., 1987; Hoyer, 2004; Srivastava & Buntine, 1995; Cichocki & Zdunek, 2007; Koren & Bell, 2011).

Matrix factorization techniques feature an easy way to incorporate prior knowledge. For example, suppose we have the initial guess of the either matrix $\mathbf{W}$ or $\mathbf{H}$. The first estimate of $\mathbf{W}$ or $\mathbf{H}$ will result in the optimal (in the least squares sense) result given the initial guess. Subsequent iterations will lead to further refinements in the previous estimates of $\mathbf{W}$ or $\mathbf{H}$. In the current scope of this data challenge, we intend to demonstrate the applicability of the above mentioned algorithms to

Input: $\mathbf{Y}$, $m$ (desired rank),$\mathbf{W}$, $\mathbf{H}$, $Q$ (stopping criteria),$\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ (regularization parameter)
Step 1(: Initialize $\mathbf{W}$, $\mathbf{H}$.
Step 2: Whilenot $Q$)

a) Update $\mathbf{W}$:=$\frac{(\mathbf{Y}\mathbf{H}^T)}{(\mathbf{H}\mathbf{H}^T + \lambda_1 I)}$;

b) $\mathbf{W} = \mathbf{W}. * (\mathbf{W} >= 0)$;

c) Update $\mathbf{H}$:=$\frac{\mathbf{W}^T\mathbf{Y}}{(\mathbf{W}^T\mathbf{W} + \lambda_2 I))}$;

d) $\mathbf{H} = \mathbf{H}. * (\mathbf{H} >= 0)$;

end
Output: $\mathbf{H}$, $\mathbf{W}$

Figure 3. Pseudo code for NMF Algorithms with alternating least squares update using both regularization parameters as shown in Equation 3.

map $\mathbf{Y}$ into a low dimensional ($e$) event space defined by the inner product of "Problem Types"-events relationship matrix $\mathbf{W}$ and events -"Case" relationship $\mathbf{H}$. Unlike test data the training data contains known ground truth for all the cases that correspond to "Problem Types" or "Nuisance Cases" and therefore we can compute $\mathbf{Y} = Y_{cp}^T$ following the logic explained in (refer Step 11) Fig. 2. If we assume that the blind sources are same as the event combination for each case, then they are essentially known to us, i.e. we can assign $\mathbf{H} = Z_{ec}$ (refer Step 10 in Fig. 2). With $\mathbf{Y}$ and $\mathbf{H}$ known the problem statement boils down to a simple estimation of $\mathbf{W}$ and hence we do not update $\mathbf{H}$ (Step 2c in Fig. 3) and set $\lambda_2 = 0$, while $\mathbf{W}$ was estimated using Step 2a (Fig. 3) with $\lambda_1 \geq 0$.

With $\mathbf{W}$ estimated we can reconstruct $\tilde{\mathbf{Y}}$ and compare it with $\mathbf{Y}$ to evaluate the prediction performance using Eqn. 1. In the feature selection stage, the relevant events $\hat{e}$ from $e$ are chosen based on their IDF scores for which the model $M_{\hat{e}}$ maximizes score prediction performance on the training data. However selecting the features using this mechanism may sometimes lead to over fitting.

### 3.4. Results and Discussion

To infer the "Problem Types", we first preprocess the test data to obtain $Z_{ec}^{tst}$ following the same logic described in Fig. 2 and then estimate $\mathbf{Y}^{tst}$=$\mathbf{W}$ $\mathbf{H}^{tst}$, where $\mathbf{H}^{tst} = Z_{ec}^{tst}$. For this experiment the regularization parameter $\lambda_1$ was tuned to $4$. In Table 1 we present the outcome of the recommender for both the training and the test set. The first column of Table 1 represents the identified "Problem Types" and the second and forth column represent the ground truth of the training and test sets respectively. Column 3 & 5 represent the percentage of the accurately predicted "Problem Types" (expressed in %) of all the train and test cases for which the ground truth is know. In the training set there are 13 unique "Problem Types" and a total of $164$ cases. However the test set has $14$ unique "Problem Types" and a total of $174$ cases for which the prediction has to be made. It should be noted that the training data does not contain any examples for $P0932$ and $P6880$ types present in the test set and the model is not expected to make any kinds to predictions on these types as they have been anyway eliminated from $F_{ces}^{tr}$ while computing $Z_{ec}^{tst}$ during the preprocessing stage. Beside $P2651$ type, the prediction accuracy for the rest of the "Problem Types" are considerably worse. The overall accuracies of the prediction on the training and the test sets are $87.5\%$ and $49.3\%$ respectively. There can be may reasons to this poor accuracy on test set, like limited examples of overall "Problem Types" as compared to "Nuisance Cases" in the training set, skewness in the examples of "Problem Types" or even over fitting due to poor selection of tuning parameters. The selection of the tuning parameters can be handled in many different ways as described in (Cichocki & Zdunek, 2007). There exists a huge scope in improving the generalized performance of the ALS NMF based prediction algorithm. It requires some experimentation with different variations of the above mentioned cost function along with an appropriate set of features and correct parameter settings before we obtain the model that best fits this data set.

## 4. CONCLUSION

This paper has demonstrated the application of a popular collaborative filtering based technique to recommend maintenance actions. In this paper we have discussed the use of standard algorithms like ALS NMF to extract mixture matrix (basis vectors) as a necessary step to predict "Problem Types". The above mentioned algorithm has demonstrated the ability to predict unknown "Problem Types" for new event sequences. Though ALS NMF based algorithm is better suited for large scale problems, a key weakness of the algorithm, however, is that it assumes that the mixing between $\mathbf{W}$ and $\mathbf{H}$ is linear. It may be the case that a nonlinear mixing have occurred. Depending on the nature of the nonlinearity, this algorithm may not correctly capture the appropriate basis factors. Another venue to explore is to inspect *2 or 3* top scoring (probable) "Problem Types" for misclassified examples from training set.

## ACKNOWLEDGMENT

This work was supported through funding from the NASA Aeronautics Research Mission Directorate, Aviation Safety Program, System-wide Safety Assurance Technologies (SSAT) Project.

## REFERENCES

Adomavicius, G., & Tuzhilin, A. (2005). Toward 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*(6), 734–749.

| "Problem Types" | Training (true) | Training (predicted %) | Test (true) | Test (Predicted %) |
|---|---|---|---|---|
| P0159 | 19 | 94.74 | 15 | 53.33 |
| P0898 | 4 | 50 | 6 | 50 |
| P1737 | 2 | 50 | 2 | 0 |
| P2584 | 53 | 86.79 | 26 | 69.23 |
| P2651 | 13 | 91.31 | 13 | 92.31 |
| P3600 | 17 | 88.24 | 20 | 70 |
| P6559 | 3 | 66.67 | 1 | 0 |
| P7547 | 6 | 50 | 4 | 75 |
| P7695 | 17 | 82.35 | 37 | 51.35 |
| P7940 | 1 | 100 | - | - |
| P9766 | 14 | 92.86 | 12 | 67.66 |
| P9965 | 2 | 50 | 5 | 40 |
| P9975 | 13 | 92.31 | 16 | 12.5 |
| P0932 | - | - | 2 | 0 |
| P6880 | - | - | 15 | 0 |
| Overall | 164 | 87.5 | 174 | 49.43 |

Table 1. In this Table we present the outcome of the recommender for both the training and the test set. The first column of represents the identified "Problem Types" and the second and forth column represent the ground truth of the training and test sets respectively. Column 3 & 5 represent the percentage of the accurately predicted "Problem Types" (expressed in %) of all the train and test cases. Each row represents a "Problem Type" and their corresponding true and predicted values. The overall statistics is shown in the last row.


Berry, M. W., Browne, M., Langville, A. N., Pauca, V. P., & Plemmons, R. J. (2006). Algorithms and applications for approximate nonnegative matrix factorization. In *Computational statistics and data analysis* (pp. 155–173).

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, *3*, 993–1022.

Cichocki, A., & Zdunek, R. (2007). Regularized Alternating Least Squares Algorithms for Non-negative Matrix/Tensor Factorization. In (pp. 793–802).

Condliff, M. K., Lewis, D. D., & Madigan, D. (1999). Bayesian mixed-effects models for recommender systems. In *In acm sigir 99 workshop on recommender systems: Algorithms and evaluation.*

Hoyer, P. (2002). Non-negative sparse coding. In *Neural networks for signal processing, 2002. proceedings of the 2002 12th ieee workshop on* (p. 557-565).

Hoyer, P. (2004, December). Non-negative matrix factorization with sparseness constraints. *J. Mach. Learn. Res.*, *5*, 1457–1469.

Koren, Y., Bell, R., & Volinsky, C. (2009, August). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37. doi: 10.1109/MC.2009.263

Koren, Y., & Bell, R. M. (2011). Advances in collaborative filtering. In *Recommender systems handbook* (p. 145-186).

Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *In nips* (pp. 556–562). MIT Press.

Papineni, K. (2001). Why inverse document frequency. In *Proceedings of the naacl* (pp. 25–32).

Srivastava, A., & Buntine, W. (1995). Data analysis of components in the optical plume of the space shuttle main engine. In *Proceedings of the aiaa.*

Su, X., & Khoshgoftaar, T. M. (2009, January). A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, *2009*, 4:2–4:2.

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, *2*(13), 37 - 52.


## BIOGRAPHIES

**Santanu Das, Ph.D.** is a member of the data science team at NASA Ames Research Center since 2006. Currently he is a research scientist at University Affiliated Research Center (UARC), a contractor of NASA Ames. His research interests are systems health management, machine learning and data driven analytics. He has over 8 years of industrial experience in designing, developing concepts, formalizing approaches to meet challenges of solving complex data problems in various domains.