South African Statistical Journal
Vol. 56, No. 2, pp. 69–89
https://doi.org/10.37920/sasj.2022.56.2.1

Algorithms for estimating the parameters of factorisation machines

E. Slabber, T. Verster and P. J. de Jongh

Centre for Business Mathematics and Informatics, North-West University, South Africa

Since the introduction of factorisation machines in 2010, it became a popular prediction technique amongst machine learners who applied the method with success in several data science challenges such as Kaggle or KDD Cup. Despite these successes, factorisation machines are not often considered as a modelling technique in business, partly because large companies prefer tried and tested software for model implementation. Popular modelling techniques for prediction problems, such as generalised linear models, neural networks, and classification and regression trees, have been implemented in commercial software such as SAS which is widely used by banks, insurance, pharmaceutical and telecommunication companies. To popularise the use of factorisation machines in business, we implement algorithms for fitting factorisation machines in SAS. These algorithms minimise two loss functions, namely the weighted sum of squared errors and the weighted sum of absolute deviations using coordinate descent and nonlinear programming procedures. Using a simulation study, the above-mentioned routines are tested in terms of accuracy and efficiency. The prediction power of factorisation machines is then illustrated by analysing two data sets.

Keywords: Factorisation machines, Fitting algorithms, Parameter estimation.

1. Introduction

When Rendle (2010) introduced factorisation machines (FMs), he described it as a new model class that combines the advantages of support-vector machines (SVMs) with those of factorisation models. According to Rendle, FMs model all interactions between predictor variables using factorised parameters that enable FMs to estimate interactions even in problems with huge sparsity where SVMs fail. Although higher order factorisation machines are defined in Rendle (2010), the models include many terms and fitting thereof is problematic (see Rendle, 2012). Therefore, in this paper, we will concentrate on fitting second order factorisation machines only. Yurochkin et al. (2017) describe a second order factorisation machine (FM) as a linear regression model which include all the predictors as well as the approximations of all the second order interactions between the predictors. Application areas of FMs, amongst others, include recommender systems (see e.g. Parsons, 2017), click-through rate prediction (Juan et al., 2016), marketing prediction problems (Juan et al., 2017), and social network prediction problems (Hong et al., 2013). We will now define factorisation machines.

Corresponding author: E. Slabber (erika.slabber@outlook.com)
MSC2020 subject classifications: 62H99.

Suppose we have observations $\{Y_n, X_{n1}, \ldots, X_{nK}\}, n = 1, \ldots, N$, on a target or response variable Y and K predictor variables $X_1, \ldots, X_K$. For the moment we will assume that the variable Y is continuous, while the predictors can be of any type, i.e. continuous, discrete, nominal or ordinal. The model equation is given by

$$Y_n = \beta_0 + \sum_{k=1}^{K} \beta_k X_{nk} + \sum_{k=1}^{K-1} \sum_{j=k+1}^{K} \left\langle \boldsymbol{\varphi}_k, \boldsymbol{\varphi}_j \right\rangle X_{nk} X_{nj} + e_n, \tag{1}$$

where β_0 is the intercept, $\beta_1, \ldots, \beta_K$ the regression coefficients, $\boldsymbol{\varphi}_k, k = 1, ..K$ a G dimensional vector of factor loadings for each variable, and e_n the error term. Here $\langle \boldsymbol{\varphi}_k, \boldsymbol{\varphi}_j \rangle = \sum_{g=1}^{G} \varphi_{kg} \varphi_{jg}$ denotes the inner product of the vectors $\boldsymbol{\varphi}_k$ and $\boldsymbol{\varphi}_j$. Note that both $\boldsymbol{\beta}$ and $\boldsymbol{\varphi}$ must be estimated and that G is an input parameter for the fitting procedure.

Second order factorisation machines relate to regression with two-way interactions in the following way. The linear regression model with all possible two-way interactions is

$$Y_n = \beta_0 + \sum_{k=1}^{K} \beta_k X_{nk} + \sum_{k=1}^{K-1} \sum_{j=k+1}^{K} \beta_{kj} X_{nk} X_{nj} + e_n, \tag{2}$$

where, as before, the β_k are the regression coefficients and the β_{kj} the coefficients of the interactions that need to be estimated. Careful examination of (1) and (2) indicates that β_{kj} in (2) is replaced by $\sum_{g=1}^{G} \varphi_{kg} \varphi_{jg}$ in (1). Rendle's key idea emanated from the well-known result that for any $(K \times K)$ positive definite matrix $\{\beta_{kj}\}$, there exists a $(K \times G)$ matrix $\{\varphi_{kg}\}$ such that $\{\beta_{kj}\} = \{\varphi_{kg}\}\{\varphi_{gj}\}$, provided that G is sufficiently large. The interaction β_{kj} is therefore approximated by a sum of simple products consisting of G factors, i.e.

$$\beta_{kj} \approx \sum_{g=1}^{G} \varphi_{kg} \varphi_{jg}. \tag{3}$$

Clearly, by choosing $\beta_{kj} = \sum_{g=1}^{G} \varphi_{kg} \varphi_{jg}$, (1) is a special case of (2). Note that the FM model in (1) has $1 + K + KG$ parameters (or coefficients) while the regression model (2) has $1 + K + K\,(K-1)\,/2$ parameters. Therefore, when $G < (K-1)/2$, an FM requires less parameters to be estimated. When the number of model parameters exceed the number of observations, two-way interaction multiple regression models are impractical to fit (see e.g. Hastie et al., 2015 or Kutner et al., 2005). This is frequently the case when dealing with recommender system problems, which are characterised by many categorical predictor variables, each having many levels (see e.g. Rendle, 2010, and Parsons, 2017). When these variables are encoded as dummy variables, the predictor variable space have many zeros leading to the problem of high sparsity. Here FMs provide an advantage through the careful selection of the number of factors G which is independent of the number of predictors. To summarise, the important difference to regression is that the effect of interaction is not modelled by an independent parameter β_{kj}, but with a factorised parameterisation $\sum_{g=1}^{G} \varphi_{kg} \varphi_{jg}$, which implies that the pairwise interactions have low rank. This allows FMs to estimate reliable parameters even in highly sparse data where standard models fail.

The linear model (2) may be extended to a generalised linear model (GLM), which allows the modelling of nominal or ordinal target variables by postulating a linear relationship between the

target and predictors, even though their underlying relationship is not linear. This is made possible by using a link function, which links the target variable to a model that is linear in its parameters. Unlike linear regression, the error distribution of the target variable need not be normally distributed but are assumed to follow an exponential family of distributions such as normal, binomial, Poisson or gamma distributions. For example, if the target variable is binary the logistic link function may be used to link the target variable to a linear model including the predictors and their two-way interactions. Such models are easy to fit using PROC GLM in SAS through maximisation of the appropriate log likelihood function. Algorithms for fitting FMs with different loss functions are discussed in Rendle (2012) and incorporated in his LibFM package. Rendle (2012) estimates the model parameters by defining a loss function and then minimising the sum of losses over the observed data. He considered squared error and logit loss, as well as L2 regularisation. LibFM, written in C, uses logit loss that may be used to fit logistic regression FM analogues of GLMs (see Pijnenburg and Kowalczyk, 2017). Note that when the underlying model is not intrinsically linear, care should be taken when interpreting the interaction effects (see Norton et al., 2004). FM extensions of GLM models, such as logistic FMs, will not be considered in this paper, but in a future paper in which we investigate alternative credit scoring modelling techniques.

Apart from LibFM, a number of FM fitting routines are available, such as PROC FACTMAC in SAS® Visual Data Mining and Machine Learning (SAS Viya), *fastFM* in Python (see Bayer, 2016) and *rsparse* an R package for statistical learning on sparse matrices (see Selivanov, 2021). In a previous paper (Slabber et al., 2021), we illustrated the use of *LibFM* and PROC FACTMAC and highlighted some of the shortcomings of these routines. For example, LibFM only outputs the predicted values of the model but not the estimated parameters, while PROC FACTMAC outputs the predicted values and the estimated parameters but has the limitation that it can only handle nominal predictor variables and an interval scaled target variable (see SAS Institute Inc., 2017, 2019). LibFM does not have these limitations and can handle any type of predictor or target variable, while PROC FACTMAC is restricted to recommender system type applications.

To make the application of FMs more accessible to business, we implement our FM fitting routines in SAS, which is widely used, especially by large corporates such as banks, insurance and telecommunication companies. We consider two algorithms for fitting FMs namely coordinate descent (CD) and nonlinear programming (NLP). The CD algorithm is implemented in PROC IML and the NLP algorithm in PROC OPTMODEL of SAS. Two loss functions are considered, namely the weighted sum of squared errors (WSSE) and the weighted sum of absolute deviations (WSAD). While WSSE is the most common loss function used to fit FMs, WSAD provides a robust alternative, since it is less sensitive to outliers. For more information on robust estimation and outlier detection, see for example Rousseeuw and Leroy (2005).

The layout of the paper is as follows. In the next section, for both loss functions, we show how the CD method may be implemented to fit FMs. The NLP implementation of these loss functions in PROC OPT MODEL is straightforward and will be mentioned briefly. In Section 3, the accuracy and efficiency of these routines are tested by means of a simulation study, and then the routines are applied to an artificially generated and real data set. Section 4 contains some concluding remarks and ideas for future research.

2. Fitting factorisation machines

Given estimators $b_0, b_1, \ldots, b_K$ and $f_{kg}, k = 1, \ldots, K, g = 1, \ldots, G$, of the parameters, we denote the corresponding estimator of Y_n by

$$\widehat{Y}_n = b_0 + \sum_{k=1}^{K} b_k X_{nk} + \frac{1}{2} \sum_{g=1}^{G} \left[\left(\sum_{k=1}^{K} f_{kg} X_{nk} \right)^2 - \sum_{k=1}^{K} f_{kg}^2 X_{nk}^2 \right], \tag{4}$$

which is equivalent (see Rendle 2010, 2012 or Slabber et al. 2021) to

$$\widehat{Y}_n = b_0 + \sum_{k=1}^{K} b_k X_{nk} + \sum_{k=1}^{K-1} \sum_{j=k+1}^{K} \sum_{g=1}^{G} f_{kg} f_{jg} X_{nk} X_{nj}.$$

We chose to use (4), because it simplified the formulation of the CD algorithm for the loss functions in (5) and (6) below. In the case of least squares, the estimators of Y_n are obtained by minimising the weighted sum of squared errors (WSSE) given by

$$WSSE = \sum_{n=1}^{N} w_n \left[Y_n - \widehat{Y}_n \right]^2 \tag{5}$$

with respect to $b_0, b_1, \ldots, b_K$ and $f_{kg}, k = 1, \ldots K\ g = 1, \ldots G$. Here $\{Y_n\}$ denotes the data on the target variable Y and $w_1, \ldots, w_N$ are non-negative weights that sum to 1. Since it is well-known that least squares estimators are sensitive to outliers (e.g. see Maronna et al., 2019), we also consider a least absolute deviation (LAD) estimator, which minimises the weighted sum of absolute deviations (WSAD). This estimator is obtained by minimising

$$WSAD = \sum_{n=1}^{N} w_n \left| Y_n - \widehat{Y}_n \right| \tag{6}$$

with respect to $b_0, b_1, \ldots, b_K$ and $f_{kg}, k = 1, \ldots K, g = 1, \ldots G$. Again $w_1, \ldots, w_N$ are non-negative weights which sum to 1.

Remarks

- Note that the loss functions in (5) and (6) are both convex.
- Minimisation of (5) can be done using standard minimisation routines. However, the objective function (6) is non-differentiable making the minimisation problem difficult.
- In the equally weighted case ($w_i = N^{-1}$ for all $i = 1, \ldots, N$), WSSE becomes mean squared error (MSE) and WSAD becomes mean absolute deviation (MAD).
- The weights w_i could be defined to produce bounded-influence type estimators by using, for example, Mallows weights (see de Jongh et al., 1988).
- The idea of robust factorisation machines is not new, and some researchers have explored the idea (see e.g. Punjabi and Bhatt, 2018 and Liu et al., 2019), but we are not aware of an implementation of the LAD estimator considered here.

2.1 CD algorithm for minimising WSSE

Suppose we have current estimates $b_0, b_1, \ldots, b_K$ and $f_{kg}, k = 1, \ldots K, g = 1, \ldots G$, which we wish to update via the CD algorithm. We begin with the intercept or bias term. Write

$$\widehat{Y}_n(0) = \sum_{k=1}^{K} b_k X_{nk} + \frac{1}{2}\sum_{g=1}^{G}\left[\left(\sum_{k=1}^{K} f_{kg}X_{nk}\right)^2 - \sum_{k=1}^{K} f_{kg}^2 X_{nk}^2\right] = \widehat{Y}_n - b_0, \tag{7}$$

that is, $\widehat{Y}_n(0)$ is (4) without the intercept term. The updated version of the intercept, say $\widetilde{b}_0$, is obtained by minimising

$$WSSE = \sum_{n=1}^{N} w_n\left[Y_n - \widehat{Y}_n(0) - \widetilde{b}_0\right]^2 \tag{8}$$

w.r.t. $\widetilde{b}_0$. The choice of $\widetilde{b}_0$ that minimises this expression is

$$\widetilde{b}_0 = \sum_{n=1}^{N} w_n[Y_n - \widehat{Y}_n(0)] = \sum_{n=1}^{N} w_n[Y_n - \widehat{Y}_n + b_0] = b_0 + \sum_{n=1}^{N} w_n[Y_n - \widehat{Y}_n]. \tag{9}$$

Next consider b_j and write

$$\widehat{Y}_n(j) = b_0 + \sum_{k\neq j}^{K} b_k X_{nk} + \frac{1}{2}\sum_{g=1}^{G}\left[\left(\sum_{k=1}^{K} f_{kg}X_{nk}\right)^2 - \sum_{k=1}^{K} f_{kg}^2 X_{nk}^2\right] = \widehat{Y}_n - b_j X_{nj}, \tag{10}$$

so $\widehat{Y}_n(j)$ is (4) without $b_j X_{nj}$.

Minimising

$$WSSE = \sum_{n=1}^{N} w_n\left[Y_n - \widehat{Y}_n(j) - \widetilde{b}_j X_{nj}\right]^2 \tag{11}$$

w.r.t. $\widetilde{b}_j$, we obtain the updated version of the j-th regression coefficient $\widetilde{b}_j$.To minimise this with respect to $\widetilde{b}_j$, note that we have a regression problem in which the X_{nj} are regressed on the residuals $Y_n - \widehat{Y}_n(j) = Y_n - \widehat{Y}_n + b_j X_{nj}$ and the minimising choice of $\widetilde{b}_j$ is given by

$$\widetilde{b}_j = b_j + \frac{\sum_{n=1}^{N} w_n[Y_n - \widehat{Y}_n]X_{nj}}{\sum_{n=1}^{N} w_n X_{nj}^2}. \tag{12}$$

Next, we consider f_{jh}. To isolate the contribution of f_{jh} to (5) is somewhat more complicated but manipulation leads to

$$\widehat{Y}_n = b_0 + \sum_{k=1}^{K} b_k X_{nk} + \frac{1}{2}\sum_{g\neq h}^{G}\left[\left(\sum_{k=1}^{K} f_{kg}X_{nk}\right)^2 - \sum_{k=1}^{K} f_{kh}^2 X_{nk}^2\right] + \frac{1}{2}\left[\left(\sum_{k\neq j}^{K} f_{kh}X_{nk}\right)^2 - \sum_{k\neq j}^{K} f_{kh}^2 X_{nk}^2\right] + f_{jh}X_{nj}\sum_{k\neq j}^{K} f_{kh}X_{nk}.$$

Denoting the sum of all but the last term by $\widehat{Y}_n(j,h)$ we get

$$\widehat{Y}_n = \widehat{Y}_n\,(j,h) + f_{jh}Z_n(j,h),$$

with

$$Z_n\,(j,h) = X_{nj}\sum_{k\neq j}^{K} f_{kh}X_{nk}. \tag{13}$$

Again $\widehat{Y}_n(j,h)$ is (4) without the term $f_{jh}Z_n\,(j,h)$. Let $\widetilde{f}_{jh}$ denote the updated version of f_{jh}, then it must minimise

$$WSSE = \sum_{n=1}^{N} w_n\left[Y_n - \widehat{Y}_n(j,h) - \widetilde{f}_{jh}Z_n(j,h)\right]^2. \tag{14}$$

To minimise this equation with respect to $\widetilde{f}_{jh}$, note that we again have a regression problem in which $Z_n(j,h)$ is regressed on the residuals $Y_n - \widehat{Y}_n\,(j,h) = Y_n - \widehat{Y}_n + f_{jh}Z_n(j,h)$ and the minimising choice of $\widetilde{f}_{jh}$ is given by

$$\widetilde{f}_{jh} = f_{jh} + \frac{\sum_{n=1}^{N} w_n[Y_n - \widehat{Y}_n]Z_n(j,h)}{\sum_{n=1}^{N} w_n Z_n^2(j,h)}. \tag{15}$$

Thus, the coordinate descend algorithm for minimising WSSE proceeds as follows:

(a) Start with some initial choice of the estimates (more below).

(b) Keeping the b_k and the f_{kg} at their values, compute a new value for b_0 from (9).

(c) Then, keeping all else fixed cycle through $j = 1,\ldots,K$, computing new values for b_j from (12).

(d) Again, keeping all else fixed, cycle through $j = 1,\ldots,K$, $h = 1,\ldots,G$, computing new values for f_{jh} from (15).

(e) Repeat steps (b)–(d) until convergence, giving the final CD estimates.

Regarding the choice of initial estimates, we can take $b_k = 0$ for $k = 1,\ldots,K$, or as the regression coefficient estimates resulting from a multiple regression fit to the data. We cannot set the f_{jh} equal to 0 since we then have $Z_n(j,h) = 0$ by (13), and (15) fails to produce an update. Hence, something else is needed and in our experiments we followed Rendle (2012) by choosing the initial f_{jh} randomly from a standard uniform distribution denoted by U(0,1) or a standard normal distribution denoted by N(0,1).

2.2 CD algorithm for minimising WSAD

In this subsection, the algorithm for WSAD is described. It follows the same steps as for WSSE except that the squared deviations in (8), (11) and (14) are replaced by absolute deviations.

Taking $\widehat{Y}_n\,(0)$, $\widehat{Y}_n\,(j)$, $\widehat{Y}_n(j,h)$ and $Z_n(j,h)$ as defined before, we have current estimates b_0, $b_1,\ldots,b_K$ and f_{kg}, $k = 1,\ldots,K$, $g = 1,\ldots,G$ which we wish to update via the CD algorithm.

If $\widetilde{b}_0$ denotes the updated version of the intercept, then

$$WSAD = \sum_{n=1}^{N} w_n \left| Y_n - \widehat{Y}_n(0) - \widetilde{b}_0 \right|$$

must be minimised w.r.t. $\widetilde{b}_0$. The choice of $\widetilde{b}_0$ that minimises this expression is difficult to obtain, and we will use linear programming software to solve for $\widetilde{b}_0$.

If $\widetilde{b}_j$ denotes the updated version of the jth regression coefficient, then $\widetilde{b}_j$ must minimise

$$WSAD = \sum_{n=1}^{N} w_n \left| Y_n - \widehat{Y}_n(j) - \widetilde{b}_j X_{nj} \right|.$$

To minimise this with respect to $\widetilde{b}_j$, note that we just have a regression problem in which the X_{nj} are regressed on the residuals $Y_n - \widehat{Y}_n(j) = Y_n - \widehat{Y}_n + b_j X_{nj}$ and the minimising choice of $\widetilde{b}_j$ is again not straightforward and will be solved by using linear programming.

If $\widetilde{f}_{jh}$ denotes the updated version of f_{jh}, it must be chosen to minimise

$$WSAD = \sum_{n=1}^{N} w_n \left| Y_n - \widehat{Y}_n(j,h) - \widetilde{f}_{jh} Z_n(j,h) \right|.$$

To minimise this with respect to $\widetilde{f}_{jh}$, note that we again have a regression problem in which $Z_n(j,h)$ is regressed on the residuals $Y_n - \widehat{Y}_n(j,h) = Y_n - \widehat{Y}_n + f_{jh} Z_n(j,h)$ and the minimising choice of $\widetilde{f}_{jh}$ will again be solved by linear programming.

The coordinate descend algorithm for minimising WSAD follows the same steps as the WSSE routine, but by substituting the WSSE parameter updating equations (9), (12) and (15) with the linear programming equivalent. The initial estimates may be obtained as before, however one might consider substituting the multiple regression coefficient estimates with the estimates obtained from a LAD regression.

2.3 Implementation of the algorithms

Both the above-mentioned algorithms were programmed in SAS using PROC IML and we used PROC LPSOLVE for solving the linear programming problems. See SAS Institute Inc. (2017) for more information on PROC IML and PROC LPSOLVE. As convergence criterion we used the mean squared differences of the change in adjusted coefficients at each subsequent iteration and as a measure of the prediction performance we used the mean squared error. Although CD does not guarantee a global minimum, the loss functions considered here are both convex having a unique minimum, and therefore concern regarding the convergence of the algorithms is alleviated. We will refer to the WSSE CD implementation as the WSSE CD routine and to the WSAD CD implementation as the WSAD CD routine.

Both loss functions were also implemented using the general nonlinear programming (NLP) solver implemented in SAS PROC OPTMODEL. Implementation is straight forward since it only requires the specification of the FM model and the loss function to be minimised. PROC OPTMODEL is a versatile solution giving the programmer the benefit of powerful tried and tested built-in optimisation routines as well as intelligent selection of the starting values for the parameters (see SAS Institute

Inc., 2014). We will refer to these OPTMODEL implementations as the WSSE NLP and WSAD NLP routines. Note that a drawback of these routines is that it requires the SAS/OR license at a substantial cost. Because of this high price tag, the CD algorithms offer a useful alternative since it can be implemented in any programming language. Using both the CD and NLP implementations allow us to validate the numerical accuracy of the two optimisation implementations.

Note that all analyses were performed on a laptop computer (i7 processor, CPU = 1.8 GHz).

3. Examples

In this section, we firstly test the accuracy of our algorithms by means of a simulation study, and secondly fit FMs to an artificially constructed ratings data set and lastly to the well-known Boston house price data set. Note that we used equally weighted loss functions in all calculations performed in this section.

3.1 Testing the accuracy of the fitting routines

Consider the FM model given in (1). We will first fit a 1-factor and then a 2-factor FM on two artificially generated data sets using the WSSE CD routine and secondly, test the accuracy of all the above-mentioned routines by means of a simulation study.

Data set 1

For the first dataset we took $K = 5$ predictors and $G = 1$ factor. The true values chosen for the parameters are shown in Table 1. $N = 1000$ observations were generated with all 5 predictors taken to be independent and identically distributed according to the standard normal distribution N(0,1) and the error term was taken to be N(0,0.25) distributed. Using equally weighted WSSE, the resulting estimates are shown in Table 1.

Except for the intercept, the estimates of the regression coefficients are quite close to their true values. This is also true for the estimates of the factor loadings provided that we change their signs. From (1) we note that replacing all the φ_k by $-\varphi_k$ yields the exact same model so that changing the signs of the estimates is valid. The MSE of the fitted model is 0.3225, which is nearly 30% larger than the true error variance of 0.25. The discrepancies noted in the estimates of the intercept and MSE could be due to sampling error, and this will be investigated in more detail in the simulation study.

Table 1. WSSE CD fit on data set 1.

Parm β	True value	Estimate	Parm φ	True value	Estimate
β_0	0.5	0.29			
β_1	0.8	0.79	φ_{11}	0.5	−0.52
β_2	0.3	0.27	φ_{21}	−0.6	0.54
β_3	0.2	0.17	φ_{31}	0.2	−0.18
β_4	0.0	0.00	φ_{41}	0.0	0.00
β_5	0.0	0.03	φ_{51}	0.0	−0.03

Data set 2

For the second data set we took $K = 8$ predictors and $G = 2$ factors. The true values chosen for the parameters are shown in Table 2 below. $N = 2500$ observations were generated with all 8 predictors taken to be standard normal variates and the error term as before. The estimates are also shown in the table. Again, except for the intercept, the estimates of the regression coefficients are very close to their true values. The MSE of the fitted model is 0.3072 and about 22% larger than the true error variance of 0.25 Again these discrepancies will be studied in the simulation study below.

Note that the estimates of the factor loadings in this example may give the impression that the estimated factor loadings are always close to the true values up to a sign change. When fitting FMs to other data sets we found that the estimated factor loadings are not always close to the true values, but that the inner product of the estimated factor loadings is always close to the inner product of the true factor loadings.

Simulation study

The objective of this study is to investigate the accuracy and efficiency of the routines implemented. Assuming the model parameters as given in Table 2, we simulate 100 data sets of 2500 observations each, by drawing the errors in model (1) from a contaminated normal distribution. Here the errors were drawn with probability 0.9 from an N(0,0.25) distribution and, in order to simulate the presence of outliers, with probability 0.1 from an N(0,25) distribution. This was done to check whether the WSAD routines outperform the WSSE routines when outliers are present. Since the estimates of the factor loadings sometimes vary in terms of sign and closeness to the true factor loadings, we decided to report the accuracy of the factor estimates in terms of the deviance of the true inner product of the factor loadings with that of the associated estimated inner product. For each combination of predicted variables, the true inner product is given in Table 3. Note from (3) that these values can also be viewed as the true implied interaction coefficients.

Starting values for the parameters of the NLP routines were not specified and we left that for PROC OPTMODEL to decide. For the CD routines, the starting values for the coefficients of the predictor variables and intercept, were obtained by fitting a least squares (LS) regression for WSSE and a LAD

Table 2. WSSE CD fit on data set 2.

Parm β	**True value**	**Estimate**	**Parm** φ	**True value**	**Estimate**	**Parm** φ	**True value**	**Estimate**
β_0	0.5	0.26						
β_1	0.8	0.80	φ_{11}	0.5	0.45	φ_{12}	1.0	1.03
β_2	0.3	0.30	φ_{21}	−0.6	−0.62	φ_{22}	0.0	−0.01
β_3	0.2	0.20	φ_{31}	0.2	0.18	φ_{32}	0.0	0.02
β_4	0.0	0.00	φ_{41}	0.0	−0.03	φ_{42}	0.0	−0.01
β_5	0.0	0.01	φ_{51}	0.0	0.00	φ_{52}	0.5	0.49
β_6	0.1	0.10	φ_{61}	0.0	−0.01	φ_{62}	0.5	0.50
β_7	0.2	0.20	φ_{71}	0.0	0.03	φ_{72}	−0.5	−0.50
β_8	0.9	0.90	φ_{81}	1.0	0.95	φ_{82}	0.5	0.53

Table 3. True inner product of factor loadings.

	X_2	X_3	X_4	X_5	X_6	X_7	X_8
X_1	−0.30	0.10	0.00	0.50	0.50	−0.50	1.00
X_2		−0.12	0.00	0.00	0.00	0.00	−0.60
X_3			0.00	0.00	0.00	0.00	0.20
X_4				0.00	0.00	0.00	0.00
X_5					0.25	−0.25	0.25
X_6						−0.25	0.25
X_7							−0.25

regression for WSAD. In both cases the starting values for the factor loadings were generated from a standard uniform distribution.

The results of the simulation study are given in Tables 4, 5, 6 and 7. The performance of the routines in terms of the number of iterations until convergence and time taken to converge is given in Table 4. Note that the convergence criteria were calculated for each data set and the table contains the summary statistics (average, standard deviation, minimum and maximum) obtained over 100 repetitions.

As expected, when compared to the more intricate WSAD routines, the number of iterations needed until convergence are less for the WSSE routines. Also, the time taken to converge, measured in hours (h), minutes(m) and seconds(s), clearly shows the superiority of the NLP implementations in terms of computational efficiency. Interestingly, compared with the WSAD NLP routine, the WSAD CD routine needs fewer iterations to converge, but takes much longer.

We calculated the MSE for each generated data set and then averaged the MSEs over simulation runs. As seen in Table 5, the average MSEs obtained by the four routines are within 2% of the theoretical variance of 2.725. This alleviates the concern we had in the previous section about the MSE estimate not being close to the true value.

Note that the two WSSE routines give the same answers, as do the two WSAD routines. Closer

Table 4. Summary statistics of performance criteria.

		Average	**Standard deviation**	**Minimum**	**Maximum**
Number of iterations	WSSE CD	32.03	9.4	14	62
	WSSE NLP	9.32	1.18	7	13
	WSAD CD	98.15	36.54	19	240
	WSAD NLP	191.27	115.89	105	732
Time to converge	WSSE CD	7.95s	2.70s	3.36s	16.92s
	WSSE NLP	0.15s	0.03s	0.11s	0.28s
	WSAD CD	17m53.84s	1h12m16.15s	2m44.07s	12h15m40.64s
	WSAD NLP	6.54s	4.20s	3.22s	31.78s

Table 5. Summary statistics of mean squared prediction error.

	Average	**Standard deviation**	**Minimum**	**Maximum**
WSSE CD	2.69	0.23	2.20	3.28
WSSE NLP	2.69	0.23	2.20	3.28
WSAD CD	2.71	0.24	2.21	3.30
WSAD NLP	2.71	0.24	2.21	3.30

Table 6. Summary statistics of the coefficient estimates of the predictor variables.

		Average of estimates				**Standard deviation of estimates**			
	True values	WSSE CD	WSSE NLP	WSAD CD	WSAD NLP	WSSE CD	WSSE NLP	WSAD CD	WSAD NLP
β_0	0.5	0.502	0.502	0.501	0.501	0.031	0.031	0.015	0.016
β_1	0.8	0.803	0.803	0.800	0.801	0.032	0.032	0.013	0.013
β_2	0.3	0.300	0.300	0.300	0.301	0.034	0.034	0.013	0.014
β_3	0.2	0.195	0.195	0.203	0.203	0.031	0.031	0.012	0.013
β_4	0.0	0.002	0.002	−0.001	0.000	0.029	0.029	0.014	0.014
β_5	0.0	0.000	0.000	0.001	0.001	0.030	0.030	0.014	0.015
β_6	0.1	0.098	0.098	0.101	0.100	0.033	0.033	0.014	0.014
β_7	0.2	0.205	0.205	0.201	0.202	0.032	0.032	0.014	0.014
β_8	0.9	0.899	0.899	0.898	0.898	0.030	0.030	0.013	0.013

Table 7. MSE of true and estimated inner products of factor loadings (WSSE NLP fit).

	X_2	X_3	X_4	X_5	X_6	X_7	X_8
X_1	0.0011	0.0005	0.0005	0.0007	0.0009	0.0010	0.0011
X_2		0.0005	0.0004	0.0007	0.0006	0.0005	0.0011
X_3			0.0000	0.0002	0.0003	0.0003	0.0009
X_4				0.0002	0.0002	0.0002	0.0007
X_5					0.0006	0.0007	0.0007
X_6						0.0007	0.0007
X_7							0.0012

inspection revealed that when the MSE values for each generated data set were compared, the two WSSE routines gave the same answers up to seven decimals and the WSAD routines the same answers up to two decimals. Of course, this should be the case as the loss functions are convex and therefore both CD and NLP converge to the same solution. Any difference observed relates to the stopping criteria, which can be adjusted to improve the similarity of the results obtained.

For each of the four routines, Table 6 contains the average and standard deviation of the coefficient estimates of the predictor variables. Again, the results obtained for both WSSE routines and for both WSAD routines are the same up to two decimal places. Closer inspection revealed that the coefficient estimates obtained for each data set were almost exactly the same, i.e. up to five decimals for the WSSE routines and up to two decimal places for the WSAD routines. Note that the accuracy with which the intercept term is estimated is close to that of the other regression coefficients, again alleviating our concern expressed when we analysed data sets 1 and 2.

The average of the coefficient estimates over simulation runs are close to the true values and no significant bias is observed. As expected, the standard deviation of the coefficient estimates of the robust WSAD fits are lower than the standard deviations obtained by the WSSE fits, indicating that robust FMs are worthwhile considering when outliers are present in a data set. Note that the MSE of the difference between the true and estimated coefficients can be calculated by adding the squared bias and squared standard deviation. In all cases this number is very small, i.e. about 0.001 or less. Table 7 contains the MSE of the difference between the estimated and true inner products of the factor loadings. We only provide the results for the WSSE NLP routine since the MSEs for the WSSE CD routine was identical and that of the WSAD routines even smaller. The above-mentioned results confirm that the four routines are numerically accurate, and that the NLP routines converge much faster than the CD routines.

Remarks

- We investigated the sensitivity of the results of the CD routines to different starting values by setting the starting values for the coefficients of the predictor variables to zero, and by generating the starting values for the factor loadings from a standard normal distribution. The results were virtually identical and led us to believe that the routines are not affected by different starting values, at least not in this study.

- We conducted a similar simulation study by generating the errors from a N(0, 0.25) distribution. As expected, the WSSE routines outperformed the WSAD routines in terms of having smaller standard deviations.

- Assuming model (2), we conducted a similar simulation study, but in this case specified the values of the interaction coefficients and not that of the factor loadings. Note that now 37 $(1 + 8 + 28)$ parameters have to be estimated whereas a 2-factor FM ($G = 2$) only requires the estimation of 25 $(1 + 8 + 16)$ parameters. Again, the results obtained were very similar to what we presented here. Therefore, the FM fitting routines did a remarkable job of estimating the coefficients of the predictor variables and factor loadings using the approximation in (3).

- Compared to the CD routines, the NLP routines are computationally much faster and give very similar results. Given space considerations, we decided to present only the results of the NLP routines for the two examples below.

3.2 Recommender system example

To study the behaviour of FMs in a recommender system context, we decided to construct a ratings data set exhibiting a structured pattern, and then distort the pattern to see whether FMs can pick up the original structure. The ratings, on a 5-point scale, are assumed to be provided by 20 users (denoted by U_1 to U_{20}) on 20 items (denoted by I_1 to I_{20}). For example, the items could be movies that were rated by viewers (users), where a low rating by a viewer indicates a poor movie and high rating an excellent one. The ratings are given in Table 8, and it should be clear that identical ratings by users have been organised in five blocks containing four users each. The ratings in this table will be referred to as the original data set since it contains the original ratings.

We then randomly removed 20% of the ratings, indicated by the grey shaded cells, and introduced 10 errors (or outliers) by changing the ratings in the black shaded cells. In each of these cells a rating of 1 was replaced with a 5 and a rating of 5 with a 1.

The data set that remains after the removal of the 20% ratings will be referred to as the incomplete data set (without outliers), and the incomplete data set containing the outliers as the incomplete data set (with outliers). Note that both the incomplete data sets do not contain identical users and, should the labels of users and items be randomised, the block structure will not be clearly visible. Our objective is now to determine whether FMs can correctly predict the missing ratings with and without the presence of outliers. Since we want to predict an ordinal rating using two nominal predictors, the relevance of model (1) may be questioned. However, nothing prevents us from applying the WSSE or WSAD FM fitting routines to this data set; in fact, in his first paper Rendle (2010) used MSE loss to fit the MovieLens data set which is a ratings data set like the one studied here. See also the last remark at the end of this section.

Before we can fit an FM to the incomplete data sets, we have to input it into an appropriate format understood by the algorithms. The data set was encoded using one hot or dummy coding (see Slabber et al., 2021 or Rendle, 2012). Using this method, the two predictor variables (users and items) escalate to 40 nominal predictors (the 20 levels of users plus the 20 levels of items). Suppose we want to fit an FM with 10 factors (FM10), then this will require the estimation of 441 ($1 + K + KG = 1 + 40 + 40(10) = 441$) parameters.

Note that the number of parameters are more than the number of observations (320, i.e. 400 minus the 20% missing values), which is typical when fitting recommender systems. However, compared to fitting a full two-way interaction regression model (821 parameters), the FM10 model results in a significant reduction in the number of parameters that need to be estimated. When G is equal to 4, the reduction in the number of parameters is a further 240, since instead of 441, only 201 parameters need to be estimated.

Once the FM model has been fitted, we obtain 320 estimated ratings $\widehat{y}_1, \ldots, \widehat{y}_{320}$ and 80 predicted ratings $\widehat{y}_{321}, \ldots, \widehat{y}_{400}$. These are all numeric values and do not correspond exactly to a particular rating category. In order to achieve this, we transform the numeric estimates and predictions to $\widehat{y}^*_1, \ldots, \widehat{y}^*_{400}$, where for $i = 1, \ldots, 400$,

$$\widehat{y}^*_i = \begin{cases} [0.5 + \widehat{y}_i] & \text{for } 0.5 \leq \widehat{y}_i < 5.5, \\ \widehat{y}_i & \text{otherwise,} \end{cases}$$

with $[x]$ the largest integer contained in x. For example, if $\widehat{y}_1 = 2.4$ then $\widehat{y}^*_1 = 2$ or if $\widehat{y}_2 = 2.6$ then

Table 8. Original ratings data set.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}	I_{13}	I_{14}	I_{15}	I_{16}	I_{17}	I_{18}	I_{19}	I_{20}
U_1	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5
U_2	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5
U_3	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5
U_4	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5
U_5	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4
U_6	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4
U_7	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4
U_8	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4
U_9	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3
U_{10}	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3
U_{11}	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3
U_{12}	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2	3	3	3	3
U_{13}	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2
U_{14}	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2
U_{15}	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2
U_{16}	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1	2	2	2	2
U_{17}	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1
U_{18}	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1
U_{19}	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1
U_{20}	2	2	2	2	3	3	3	3	4	4	4	4	5	5	5	5	1	1	1	1

$\widehat{y}_2^* = 3$. Let $y_i, i = 1, \ldots, 400$ denote the original (unchanged) ratings, then the fitted MSE is defined as $\frac{1}{320}\sum_{i=1}^{320}(\widehat{y}_i^* - y_i)^2$ and the predicted MSE as $\frac{1}{80}\sum_{i=321}^{400}(\widehat{y}_i^* - y_i)^2$.

Firstly, we fit the algorithms to the encoded incomplete data set (without outliers) for various choices of G. The results of the NLP WSSE and WSAD FM fits are given in Table 9 for different choices of factors $G = 2, 3, 4, 6, 8$ and 10. In addition to the fitted MSE and predicted MSE, we also counted the number of estimated and predicted transformed ratings that were unequal to the value of the original ratings and present the results under the headings 'Incorrectly fitted' and 'Incorrectly predicted' respectively. This, together with the run time of the algorithms are given in Table 9 for the NLP implementation.

The results show that, when using less than 4 factors, both the WSSE and WSAD algorithms struggle to fit the data set. Also, the run time of the WSAD routines is long, especially for smaller choices of G. As far as model selection is concerned, one would always prefer a model that generalises well and does not over-fit the data set. An indication of overfitting, and therefore poor generalisation, would be a very small fitted MSE (overfitting) and a large predicted MSE (poor generalisation). For an in-depth discussion of these topics the interested reader is referred to the excellent texts by Hastie et al. (2009), Hastie et al. (2015) and Efron and Hastie (2016). When 4 or more factors are specified, both the WSSE and WSAD algorithms obtain perfect fits, with the 4-factor models being the parsimonious choice. It is interesting to note that the untransformed estimated and predicted ratings, for the perfect fitting models, were all within one hundredth of the corresponding original ratings.

An obvious question would be to determine how many missing values will be needed before the model fails to pick up the structure in the data. Although this can be researched further, it remains remarkable how well the model is able to predict. Part of the success may be attributed to the fact that the interactions appear as inner products in (1), hence allowing the estimation of each component to "borrow strength" from other components (see Rendle, 2010, 2012; Slabber et al., 2021).

Table 9. WSSE and WSAD FM fits on the incomplete data set (without outliers).

	G	Fitted MSE	Predicted MSE	Incorrectly fitted	Incorrectly predicted	Run time
WSSE	2	0.672	1.013	206	60	0.58s
	3	0.328	0.675	105	45	0.01s
	4	0.000	0.000	0	0	1.36s
	6	0.000	0.000	0	0	0.01s
	8	0.000	0.000	0	0	7.17s
	10	0.000	0.000	0	0	11.43s
WSAD	2	Did not converge within 10 hours				
	3	0.700	0.942	18	6	1h18m17.82s
	4	0.000	0.000	0	0	1h25m1.43s
	6	0.000	0.000	0	0	2m15.30s
	8	0.000	0.000	0	0	1m50.22s
	10	0.000	0.000	0	0	14m3.06s

Secondly, for $G = 4$ and $G = 8$, we fit the algorithms to the encoded incomplete data set with outliers. In Table 10 we present the fitted and predicted MSE of the WSSE and WSAD fits. Again, we present the number of incorrectly fitted and predicted ratings and the run time of the routines. The last column contains the number of outliers that are corrected, meaning that the estimated transformed rating is equal to the original rating and not the value of the outlier.

The results in Table 10 show that the best model is WSAD FM4, since it corrected 9 outliers and gave only one incorrect prediction and 3 incorrect estimates. When observing the estimated and predicted ratings of the WSAD FM4 fit (see the first remark at the end of this section), it turns out that for user U_9 the estimated transformed ratings for I_5, I_6, and I_7 as well as the predicted rating for I_8 are all equal to 2 (untransformed rating estimate 1.92), instead of the original ratings of 5 (refer to Table 8). So, WSAD FM4 only failed in the above-mentioned 4 cases, therefore estimating and predicting the original ratings in all other cells correctly. When we examined the coefficient estimates of U_9 (see the first remark at the end of this section), we noticed that the vector of coefficient estimates of U_9, which should ideally be the same as that of U_{10}, U_{11} and U_{12}, does not fit the general block structure as if the algorithm had difficulty deciding whether U_9 is closer to U_8 or U_{10}. Interestingly, the coefficient estimates for items I_5, I_6, I_7,and I_8 are almost exactly the same. Consider Table 8 and observe the location of the outlier (U_9's rating on I_7, which is now 1 instead of 5) and the missing value (U_9's rating on I_8). The two cells find themselves on the edge of a block originally rated as 5s and adjacent to three other blocks originally rated as 1s (above), 2s (above and to the right), and 1s (to the right). The FM clearly has difficulty deciding to which of the four blocks the four problematic cells belong, and therefore the transformed estimated ratings of 2 is understandable.

When comparing the 8-factor WSSE FM fit with that of the 4-factor WSSE FM fit, the much higher predicted MSE of the former might seem strange given the much lower number of incorrect predictions. Closer inspection revealed that a few very large prediction errors caused the high predicted MSE. The 8-factor fits of both WSSE and WSAD seem to overfit since they predict poorly, but fit the data well. This was confirmed when we inspected the estimated ratings for the non-corrected outliers. In all cases the transformed estimated ratings were equal to the value of the outliers.

Remarks

- Due to space considerations, we do not provide the results of the various FM fits. These results include the coefficient and factor loading estimates as well as the estimated and predicted ratings. These results are available from the first author upon request.

Table 10. WSSE and WSAD FM fits on the incomplete data set (with outliers).

	G	Fitted MSE	Predicted MSE	Incorrectly fitted	Incorrectly predicted	Run time	Outliers corrected
WSSE	4	1492.867	18 588.896	269	68	1.21s	0
	8	0.500	224 181.994	10	13	1m59.64s	0
WSAD	4	0.084	0.113	3	1	40m21.64s	9
	8	0.325	1.720	7	7	3h36m31.67s	4

- It should be clear from the discussion here that one should consider various choices of G, before selecting the final model.
- When we applied PROC FACTMAC to the ratings data set, the results were similar to the WSSE routines, which is expected because PROC FACTMAC also employs the MSE loss function.
- As stated previously, the applicability of the FM model in (1) to this prediction problem might be questioned. Because the target variable is ordinal, it might be more appropriate to consider an FM model formulated and fitted analogous to an ordinal logistic regression. However, it should be clear that the FM models fitted in this section perform remarkably well.

3.3 Boston housing data set

To test FMs on a real-world data set, the well-known publicly available Boston housing data set is used. This data set was first used by Harrison Jr and Rubinfeld (1978) in a study regarding the impact of air pollution (as measured by the square of nitrogen oxide concentration) and 12 other explanatory variables on the price of owner-occupied homes. The data set is a sample of 506 observations on census tracts in the Boston Standard Metropolitan Statistical Area (SMSA) in 1970. For a detailed discussion see Belsley et al. (1980) and the more recent paper by Zhang (2008).

The set, which was recently updated and corrected, is freely available on various websites (see e.g. Perera, 2018). The data set consists of the price variable relevant to the analysis (*lmv* the logarithm of the median value of owner-occupied homes) and 13 explanatory variables (*crim*, *zn*, *indus*, *chas*, *nox*, *rm*, *age*, *dis*, *rad*, *tax*, *ptratio*, *black*, *lstat*). Here *crim* is the per capita crime rate by town, *zn* the proportion of a town's residential land zoned for lots greater than 25000 square feet, *indus* the proportion of non-retail business acres per town, *chas* Charles River dummy variable with value 1 if tract bounds on the Charles River, *nox* the nitric oxide concentration (parts per hundred million) squared, *rm* the average number of rooms per dwelling, *age* the proportion of owner-occupied units built prior to 1940, *dis* the logarithm of the weighted distances to five employment centres in the Boston region, *rad* the logarithm of the index of accessibility to radial highways, *tax* the full-value property tax rate per \$10 000, *ptratio* the pupil-teacher ratio by town, *black* is $1000(Bk - 0.63)^2$ where Bk is the proportion of blacks in the population and *lstat* the logarithm of the proportion of the population that is of lower status.

Because of the scale differences among the predictor variables, and because we are interested in identifying interaction effects amongst the predictor variables, we standardised all the predictor variables. See for example Frost (2019) on the importance of standardisation in regression analysis when the model contains polynomial and/or interaction terms. The importance of standardisation was not mentioned previously, because the predictors in the ratings data set are binary (dummy variables) and the predictors in the simulation study are standard normal variates, therefore not requiring standardisation.

To test the performance of FMs on this data set, we split the set in a training and test data set. As is done frequently in data competitions (see e.g. Mandav, 2021), we selected an 80% random sample from the full data set as the training set and the remaining observations as the test set. We then fitted several models on the training data set and tested the prediction performance on the test set. The models considered were a multiple regression, a multiple regression including all two-way

Table 11. Boston data: Model comparison on training and test data set.

Model	Parameters	Fitted RMSE	Predicted RMSE
Multiple regression	13	4.794	4.324
Regression (incl. two-way interactions)	92	2.467	4.008
WSSE FM2	40	3.102	3.445
WSAD FM2	40	3.624	3.381
WSSE FM4	66	2.742	5.024
WSAD FM4	66	3.133	4.927

interactions, and 2- and 4-factor (WSSE and WSAD) FMs. SAS PROC GLM was used to fit the regressions and we present the results of the NLP FM fits in Table 11. Again, the results of the CD routines were similar to that of the NLP routines, but again it took a much longer time to converge. Since the data competitions usually measure performance using root mean square error (RMSE) we report the fitted and predicted RMSE and not the MSE as was done previously.

Note that the models in Table 11 are nested within each other, therefore the smaller model will always have the larger fitted RMSE. For example, as seen in Table 11, the fitted RMSE obtained by the two-way interaction regression (the most complex model having 92 parameters) is smallest; while the fitted RMSE obtained by the multiple regression (the simplest model having 13 parameters), is the largest. As far as prediction is concerned, WSAD FM2 has the lowest predicted RMSE, closely followed by WSSE FM2. The good performance of the 2-factor FMs suggests the presence of interaction effects between the predictor variables, and the superior performance of WSAD FM2 suggests the presence of outliers. The latter has been confirmed by several studies (see e.g. Belsley et al., 1980).

To investigate the presence of interaction effects, we studied the significant interaction effects as obtained from the two-way interaction regression model. The model identified significant interaction effects between variables *crim* and *chas*, *crim* and *rm*, *rm* and *lstat*. The fact that significant interaction effects are present explains the better performance of the FM2 model. The predicted RMSE of 3.381 obtained by WSAD FM2 and 3.445 obtained by WSSE FM2, compare well with the best performers on the Kaggle leader boards; see Kaggle (2021). The leader boards list two scores based on RMSE, one based on 70% of the test data (top scores 3.72, 3.93 and 3.94) and the other one on 30% of the test data (top scores 3.69, 3.89 and 3.94). Although our scores are not directly comparable it has lowest predicted RMSE.

4. Conclusion

Assuming a factorisation machine model formulation, we minimise two well-known loss functions by using coordinate descent and non-linear programming. The coordinate descent routines were implemented in SAS PROC IML and the nonlinear programming routines in SAS PROC OPTMODEL. We demonstrated that these routines are adequate for fitting factorisation machine models and that these routines may provide a useful addition to the statistician's model building toolkit. Furthermore, the implementation in SAS might stimulate the application of FMs to many prediction problems in business. We are currently working on extending our routines to include regularisation and are testing

the performance of the routines on large data sets. Also, we are doing research on the applicability of FMs in a credit scoring context, using an FM formulation analogous to logistic regression.

Acknowledgments. The authors acknowledge that this research benefited from valuable contributions by Prof. Hennie Venter and Prof. Tertius de Wet. A detailed and extremely valuable report by a referee assisted us to substantially improve the presentation of the paper. This work is based on research supported in part by the Department of Science and Innovation (DSI) of South Africa. The grant holder acknowledges that opinions, findings and conclusions or recommendations expressed in any publication generated by DSI-supported research are those of the authors and that the DSI accepts no liability whatsoever in this regard.

References

Bayer, I. (2016). fastfm: A library for factorization machines. *Journal of Machine Learning Research*, **17**, 6393–6397.

Belsley, D. A., Kuh, E., and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. Wiley, New York, NY.

de Jongh, P. J., de Wet, T., and Welsh, A. H. (1988). Mallows-type bounded-influence-regression trimmed means. *Journal of the American Statistical Association*, **83**, 805–810.

Efron, B. and Hastie, T. (2016). *Computer Age Statistical Inference: Algorithms, Evidence, and Data Science*. Cambridge University Press, Cambridge.

Frost, J. (2019). *Introduction to Statistics: An Intuitive Guide for Analyzing Data and Unlocking Discoveries*. Statistics by Jim Publishing, State College, PA.

Harrison Jr, D. and Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, **5**, 81–102.

Hastie, T., Tibshirani, R., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, NY.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). *Statistical Learning with Sparsity; The Lasso and Generalizations*. CRC Press, Boca Raton, FL.

Hong, L., Doumith, A. S., and Davison, B. D. (2013). Co-factorization machines: Modeling user interests and predicting individual decisions in Twitter. *In Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. 557–566.

Juan, Y., Lefortier, D., and Chapelle, O. (2017). Field-aware factorization machines in a real-world online advertising system. *In Proceedings of the 26th International Conference on World Wide Web Companion*. 680–688.

Juan, Y., Zhuang, Y., Chin, W.-S., and Lin, C.-J. (2016). Field-aware factorization machines for CTR prediction. *In Proceedings of the 10th ACM Conference on Recommender Systems*. 43–50.

Kaggle (2021). House price prediction with Boston housing dataset.
URL: *https://www.kaggle.com/c/house-price-prediction-with-boston-housing-dataset*

Kutner, M. H., Nachtsheim, C. J., Neter, J., and Li, W. (2005). *Applied Linear Statistical Models*. McGraw-Hill Irwin, New York, NY.

Liu, C., Zhang, T., Li, J., Yin, J., Zhao, P., Sun, J., and Hoi, S. C. H. (2019). Robust factorization

machine: A doubly capped norms minimization. *In Proceedings of the 2019 SIAM International Conference on Data Mining*. SIAM, 738–746.

Mandav, H. (2021). Train and test data split for ML models.
URL: *https://medium.com/mlearning-ai/train-and-test-data-split-152bad0afbb2*

Maronna, R. A., Martin, R. D., Yohai, V. J., and Salibián-Barrera, M. (2019). *Robust Statistics: Theory and Methods (with R)*. Second edition. Wiley, Hoboken, NJ.

Norton, E. C., Wang, H., and Ai, C. (2004). Computing interaction effects and standard errors in logit and probit models. *The Stata Journal*, **4**, 154–167.

Parsons, N. (2017). Factorization machines for recommendation systems.
URL: *https://getstream.io/blog/factorization-recommendation-systems*

Perera, P. (2018). The Boston housing dataset.
URL: *https://www.kaggle.com/prasadperera/the-boston-housing-dataset/data*

Pijnenburg, M. and Kowalczyk, W. (2017). Extending logistic regression models with factorization machines. *In International Symposium on Methodologies for Intelligent Systems*. Springer, 323–332.

Punjabi, S. and Bhatt, P. (2018). Robust factorization machines for user response prediction. *In Proceedings of the 2018 World Wide Web Conference*. 669–678.

Rendle, S. (2010). Factorization machines. *In 2010 IEEE International Conference on Data Mining*. 995–1000.

Rendle, S. (2012). Factorization machines with LibFM. *ACM Transactions on Intelligent Systems and Technology*, **3**, 1–22.

Rousseeuw, P. J. and Leroy, A. M. (2005). *Robust Regression and Outlier Detection*. Wiley, New York, NY.

SAS Institute Inc. (2014). *SAS/OR® 13.2 User's Guide: Mathematical Programming*. SAS Institute Inc., Cary, NC.
URL: *https://support.sas.com/documentation/onlinedoc/or/132/optmodel.pdf*

SAS Institute Inc. (2017). *SAS® Visual Data Mining and Machine Learning 8.2: Procedures*. SAS Institute Inc., Cary, NC.
URL: *https://documentation.sas.com/api/docsets/casml/8.2/content/casml.pdf*

SAS Institute Inc. (2019). *SAS® Visual Data Mining and Machine Learning 8.5: Procedures*. SAS Institute Inc., Cary, NC.
URL: *https://documentation.sas.com/api/docsets/casml/8.5/content/casml.pdf*

Selivanov, D. (2021). *rsparse: Statistical Learning on Sparse Matrices*. R package version 0.5.0.
URL: *https://CRAN.R-project.org/package=rsparse*

Slabber, E., Verster, T., and de Jongh, R. (2021). Advantages of using factorisation machines as a statistical modelling technique. *South African Statistical Journal*, **55**, 125–144.

Yurochkin, M., Nguyen, X., and Vasiloglou, N. (2017). Multi-way interacting regression via factorization machines. *In Proceedings of the 31st Conference on Neural Information Processing Systems*. 2595–2603.

Zhang, T. (2008). Adaptive forward-backward greedy algorithm for sparse learning with linear

models. *Advances in Neural Information Processing Systems*, **21**, 1921–1928.

Manuscript received 2021-09-28, revised 2021-11-06, accepted 2022-07-26.