DATA 612 Project 3: Matrix Factorization Methods

Albert Gilharry

June 20, 2019

```
library(dplyr)
library(ggplot2)
library(recommenderlab)
```

Data

For this assignment I will used the recommender system from the previous project that used the Jester data set 1 from http://eigentaste.berkeley.edu/dataset/. The data set contains anonymous ratings from 24,983 users who have rated 36 or more jokes. The Ratings are real values ranging from -10.00 to +10.00. The RMSE from the using item-based and user-based collaborative filtering were 4.5869 and 4.5545 respectively. We will now build recommenders using Single Value Decomposition to see if there is any improvement in accuracy.

Singular Value Decomposition (SVD) is a common method used to reduce the dimensionality of data. This should effectively allow us to represent the main features of the data in a manner that will require less storage and processing. This data set has 2.5 million ratings and may benefit from SVD.

```
# set seed for reproducibility
set.seed(100)
# load data
data <- read.csv("data/jester-data-1.csv", header = FALSE)

# label items and ignore the first column
item_names <- paste0("Joke", seq(1,100))
ratings <-  dplyr::select(data, -1)
names(ratings ) <- item_names

# represent missing values as NA
ratings[ratings == 99] <- NA
ratings_matrix <- as.matrix(ratings)

# create realRatingMatrix for input to recommenderlab
sparse_smatrix <- as(ratings_matrix, "sparseMatrix")
real_matrix <- as(sparse_smatrix, "realRatingMatrix")
real_matrix
```

```
## 24983 x 100 rating matrix of class 'realRatingMatrix' with 2492917 ratings.
```

Model Evaluation

SVD based Models will be created using the `recommenderlab` package. 10-Fold cross-validation will used to train and evaluation the models. 85% of the data will be used for trainig the models and the remaining 15% will be used as the testing set on which the RMSE will be calculated. The main performance metric will be the RMSE but the MAE will also be looked at. A rating of 5 will be deemed as a good rating for the purposes of this project. For each test set user, all but 20 randomly selected ratings will be withheld for downstream evaluation. Models will be created with different tuning parameters.

```r
# initialize the evaluation scheme
eval_scheme <- evaluationScheme(real_matrix, method="split", train=0.85, k=5, goodRating=5, given=20)
```

SVD Based Recommendations

```r
# initialize list to store models
svd_models = list()

# initialize dataframe to store results
svd_df = data.frame(Method=character(),
                    K=integer(),
                    Normalize=character(),
                    Maxiter=integer(),
                    RMSE=double(),
                    MAE=double(),
                    stringsAsFactors = FALSE)

for(i in 1:20){
  # candiate parameter values
  K = sample(c(10,20,30,40,50,60,70,80,90,99), 1)
  Maxiter = sample(c(10,20,30,40,50,60,70,80,90,99), 1)
  Normalize = sample(c('center', 'z-score'), 1)

  # train model
  svd_model <- Recommender(data = getData(eval_scheme, "train"),
                           method = "SVD",
                           parameter = list(k = K, maxiter = Maxiter, normalize = Normalize) )

  # calculate and store RMSE
  prediction <- predict(svd_model, getData(eval_scheme, "known"), type="ratings")
  err <- calcPredictionAccuracy(prediction, getData(eval_scheme, "unknown"))
  svd_df[nrow(svd_df) + 1,] = list(Method="SVD",
                                   K=K,
                                   Normalize=Normalize,
                                   Maxiter = Maxiter,
                                   RMSE=as.numeric(err["RMSE"]),
                                   MAE=as.numeric(err["MAE"]))
  # add to list of models
  svd_models[i] <- svd_model
}

# results
dplyr::arrange(svd_df, RMSE)
```

```
##    Method  K Normalize Maxiter     RMSE      MAE
## 1     SVD 20   z-score      10 4.643912 3.759839
## 2     SVD 20   z-score      30 4.644071 3.759998
## 3     SVD 20   z-score      50 4.644071 3.759998
## 4     SVD 20   z-score      50 4.644071 3.759998
## 5     SVD 20   z-score      90 4.644071 3.759998
## 6     SVD 30    center      80 4.650058 3.762697
## 7     SVD 40    center      99 4.678957 3.788703
```

```
## 8     SVD 40    center      10 4.681555 3.790495
## 9     SVD 40   z-score      60 4.704532 3.816307
## 10    SVD 50    center      10 4.710329 3.818329
## 11    SVD 60    center      10 4.733928 3.844535
## 12    SVD 50   z-score      10 4.739768 3.849608
## 13    SVD 60   z-score      80 4.743557 3.865288
## 14    SVD 60   z-score      80 4.743557 3.865288
## 15    SVD 70    center      60 4.745471 3.864362
## 16    SVD 80    center      99 4.756195 3.876281
## 17    SVD 80   z-score      40 4.760083 3.880425
## 18    SVD 90   z-score      50 4.766371 3.887081
## 19    SVD 90    center      60 4.766758 3.886604
## 20    SVD 99   z-score      70 4.773409 3.894931
```

We can see the the best RMSE was 4.5721. The SVD based RMSE ranges between 4.5 and 4.7 but the variability is much higher for the user and item-based recommenders ranging from 4.5 to 6.7.

Biased Adjusted SVD Based Recommendations

SVD and other dimensionality reduction techniques can perform even better if the relationship between the users and the items can be broken down into components and modeled separately before aggregating to form a final prediction. We will now take the previous recommender and adjust for both user and item biases. This is by no means a comprehensive method. We will make a key assumption the SVD did not effectively captured and adjusted for the bias in these data.

```
rmses <- rep(0.0,length(svd_models))
maes <- rep(0.0,length(svd_models))

for (model in 1:length(svd_models)){

  svd_model <- svd_models[[model]]
  prediction <- predict(svd_model, getData(eval_scheme, "known"),type="ratingMatrix")

  # convert predictions to matrix for ease of use
  prediction_matrix <- as(prediction, "matrix")
  prediction_mean <- mean(prediction_matrix, na.rm = TRUE)

  # calculate user bias
  known_data <- as(getData(eval_scheme, "known"), "matrix")
  bias_corrected <- known_data
  raw_mean <- mean(known_data, na.rm = TRUE)

  # initialize bias vectors
  user_bias <- rep(0, nrow(known_data))
  item_bias <- rep(0, ncol(known_data))

  # calculate user bias
  for(i in 1:nrow(known_data)){
   user_bias[i] <- round( mean(known_data[i, which(!is.na(known_data[i,]))])) - raw_mean
  }

  user_bias_df <- data.frame(User = 1:nrow(known_data), Bias = user_bias)

  # calculate item bias
```

```r
  for(i in 1:ncol(known_data)){
   item_bias[i] <- round( mean(known_data[ which(!is.na(known_data[,i])), i])) - raw_mean
  }

  item_bias_df <- data.frame(Item = colnames(known_data), Bias = item_bias)

  # apply bias adjustments
  for(user in 1:nrow(known_data)){
    for(item in 1:ncol(known_data)){
      bias_corrected[user, item] <- prediction_mean + user_bias[user] + item_bias[item]
    }
  }

  bias_corrected[which(bias_corrected > 10)] <- 10
  bias_corrected[which(bias_corrected < -10)] <- -10

  bias_corrected <- as(bias_corrected, "realRatingMatrix")
  err <- calcPredictionAccuracy(bias_corrected, getData(eval_scheme, "unknown"))
  rmses[model] <- as.numeric(err["RMSE"])
  maes[model] <- as.numeric(err["MAE"])
}

svd_df$ADJUSTED_RMSE <- rmses
svd_df$ADJUSTED_MAE <- maes


# results
head(dplyr::arrange(svd_df, ADJUSTED_RMSE), 1)
```

```
##   Method  K Normalize Maxiter     RMSE      MAE ADJUSTED_RMSE ADJUSTED_MAE
## 1    SVD 20   z-score      10 4.643912 3.759839      4.530922     3.580901
```

We can see a slight improvement with the best RMSE now at 4.4975.

Summary

Overall, the SVD-based models we more stable than the user-based and item-based models because there were only slight devations in the SVD-based RMSEs. The best user-based model from the previous project resulted in an RMSE of **4.5071** and the best item-based model had an RMSE of **4.6190**. This project resulted in an RMSE of **4.4975**.

Sources

http://CRAN.R-project.org/package=recommenderlab https://cran.r-project.org/web/packages/recommenderlab/vignettes/recommenderlab.pdf

https://cran.r-project.org/web/packages/recommenderlab/vignettes/recommenderlab.pdf