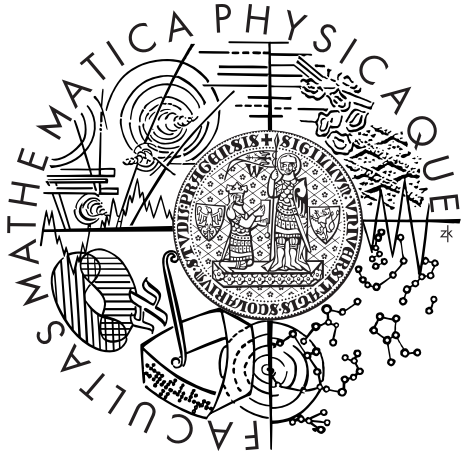
Charles University in Prague
Faculty of Mathematics and Physics

MASTER THESIS



Matej Ferenc

Content based Recommendation from Explicit Ratings

Department of Software Engineering

Supervisor of the master thesis: prof. RNDr. Peter Vojtáš, DrSc.
Study programme: Computer Science
Specialization: Software Engineering

Prague 2016

I would like to express my gratitude to my supervisor prof. RNDr. Peter Vojtáš, DrSc. for the useful comments, remarks and engagement through the learning process of this master thesis.

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

In ..... date ..... signature of the author

Název práce: Content based Recommendation from Explicit Ratings

Autor: Matej Ferenc

Katedra: Katedra softwarového inženýrství

Vedoucí diplomové práce: prof. RNDr. Peter Vojtáš, DrSc., Katedra softwarového inženýrství

Abstrakt: V této práci porovnáváme několik modelů pro predikci uživatelských preferencí. Hlavním zaměřením jsou tzv. Content Based modely, které pracují s metadatami o objektech, které doporučujeme. Ty jsou srovnány s dalšími modely, které metadata neberou do úvahy. Pro získaní výsledků používáme tři datasety a tři metriky. Cílem diplomové práce je zjistit, jak můžou metadata o uživatelích a objektech zlepšit standardní modely pro doporučení. Výsledkem ale je, že metadata sice můžou zlepšit doporučení v některých případech, záleží ale na datasetu a na metrice, která byla použita. Toto zlepšení většinou není významné.

Klíčová slova: doporučovací systémy, obsah, explicitní hodnocení

Title: Content based Recommendation from Explicit Ratings

Author: Matej Ferenc

Department: Department of Software Engineering

Supervisor: prof. RNDr. Peter Vojtáš, DrSc., Department of Software Engineering

Abstract: In the thesis we compare several models for prediction of user preferences. The focus is mainly on Content Based models which work with metadata about objects that are recommended. These models are compared with other models which do not use metadata for recommendation. We use three datasets and three metrics to get the results of recommendation. The goal of the thesis is to find out how can the metadata about the users and the objects enhance the standard recommender models. However, the result is that the metadata can enhance recommendation in some cases, but it varies by used metrics and dataset. This enhancement is not significant.

Keywords: recommender systems, content, explicit rating

Contents

1. Content based Recommendation and Explicit Ratings

1.1 Recommendation

It is already a standard for today's information systems to provide personalized information to users and about the users. The user profiles have been in use since the very beginning of computer systems but they have not always been used to alter the content that the user sees. With more powerful hardware new ways of processing the information have been created and with more users having access to computer systems there have been increasing needs to make the systems to behave differently for different users. The needs arise from both creators and users of the systems. The creators try to make their product more attractive, more usable and to gain a greater part of users on the market. The users need to find the information they need more quickly, they want the system to be smarter and to do the hard work of searching and filtering the information, so that the user does not have to spend their time working with the system.

1.2 Content

There are many kinds of information: textual, images, video and others as well as their combinations. What is meant by *content* are metadata used to describe the original information, so that no further processing is needed. The information are usually annotated by their attributes. For example, movies can be annotated by genre, actors, year of production, director and country of origin. What is not meant by *content* are data that have a certain content, but it would need further processing. For example, if we were to recommend textual data like articles, we would need to process the text to extract certain features of the article. Or if we were to recommend videos that have no further metadata, it is not really a content based recommendation.

1.3 User And Data Model

Except annotations of data, there are many other important information about the data. Most important are ratings of the data items by individual users. We call it explicit information, because the user explicitly specifies how much they like the item. Then there are implicit information. By implicit is meant every interaction of user that is not explicit. For example, number of accesses of the item by user is implicit information, as well as time spent looking at the item or number of scrolls on the page with the item. The user profile is essential for providing good recommendations for user based on the properties of data items. It is not only facts about user like their age, location and e.g. the movie genre they like most. It is also important to store user's preferences for individual items.

1.4 User Identification

The user usually identifies themselves by using their login and password and using the service or system while logged in. This is the most common case and also most useful, as the user is credibly identified. However, sometimes the user does not want to create and account, or it is not very useful, like in case they are buying a car, since it does not make a sense to create an account on such a website. In such case, there exist solutions like identifying the user using session when browsing web, or identifying them by fingerprint of their device. The fingerprints are especially useful if the user does not want to be identified, i.e. they want to surf anonymously. Another advantage is, that fingerprint identification does not use any personal data, because browser configuration is not considered to be personal data. The idea is to use HTTP, JavaScript or Flash capabilities to gain as much information about the device and browser as possible. For example, one of important characteristics of browser is user agent string. It can look like this:

Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/34.0.1847.116 Safari/537.36

However, the recommendation gives best results when the user works with the software frequently. That is why we rely on proper user identification by registration. Registration can also give us user's demographic and geolocation information. In this thesis we use user specific information, which can be extracted e.g. from the client's profile.

2. Prediction models

2.1 Introduction

In this thesis we only allow user to rate one item once. In more complex models it is also possible to allow user to rate one item multiple times and thus watch his preferences in time.

Recommender systems are traditionally categorized to two or three groups.

1. Collaborative filtering algorithms
2. Content based algorithms
3. Hybrid algorithms

The idea behind collaborative filtering algorithms (CF) is that similar users will rate the items similarly. Typical example is general user based CF: if we want to estimate rating of item i by user u we need to find most similar users to the user u. Similarity in this case is based on differences in the preference given to items rated by both users. The lower the difference, the more similar are the two users. The final estimation is usually calculated as average rating of the item i made by a few most similar users to the user u.

The content based algorithms (CB) are based on the attributes of items. The idea is that the user prefers some attributes to others. The estimate of rating of item i would be calculated as a function of ratings of the individual attributes of i.

Hybrid recommender systems are combined of the two previous ideas.

In the following sections we use two notations to describe the models: mathematical notation and pseudo algorithms. In mathematical notation we denote:

- R - set of all ratings in dataset. It can be seen as a user-item matrix which is only defined for some combinations of indexes
- U_i - all users who rated item i
- I^u - all items rated by user u
- r_i^v - observed rating of user v for item i
- $\hat{r}_i^u$ - estimated rating of user u for item i

In pseudo algorithmic notation we use a global data object called DATAMODEL which holds all the ratings and it has the following methods:

- GETPREFERENCE(*user, item*) - returns one rating from the user-item matrix:

$$R_{user,item} \tag{2.1}$$

- GETUSERS() - returns all the users. In implementation, it usually returns user IDs or references to users.

- GETITEMSRATINGS(*user*) - returns all the items and ratings for them for the *user*. In implementation it can be understood as a list of pairs of item identification and a rating value. Using mathematical notation it can be understood as a sequence of pairs where first item in pair is from I^u and the second item in pair is corresponding rating r_i^u where $i \in I_u$
- GETRATEDBYUSER(*user*) - returns all the items rated by *user*. In implementation, it usually returns user IDs or references to users. Using mathematical notation it can be understood as a set I^u.

2.2 General user based collaborative filtering

For this model it is essential to define user neighborhood and user similarity models. Once they are defined we proceed as follows:

Algorithm 1 User based Collaborative Filtering

```
1: procedure ESTIMATE(user, item)
2:     totalPreference ← 0.0
3:     totalSimilarity ← 0.0
4:     neighbors ← FIND NEIGHBORS(user)
5:     for all neighbor ← neighbors do
6:         preference ← DATAMODEL.GETPREFERENCE(neighbor, item)
7:         similarity ← USER SIMILARITY(user, neighbor)
8:         totalPreference ← totalPreference + preference * similarity
9:         totalSimilarity ← totalSimilarity + similarity
10:    end for
11:    return totalPreference / totalSimilarity
12: end procedure
```

This algorithm is universal for all definitions of models for user neighborhood and user similarity. First we find the neighbors of the specified user. This neighborhood contains a certain number of other users who are somehow most similar to the specified user. The neighborhood can contain only users who rated the specified item, but it is not required. Then for each user from the neighborhood we extract the *preference* of the neighbor for the specified item from the data model and we count the *similarity* of the neighbor and the specified user. By multiplying these two numbers we calculate weighted preference. In the end we divide weighted preference by total weight (sum of all *similarities*).

In mathematical notation the algorithm would be written as follows:

$$Ubcf(i,u) = \frac{\sum_{n \in Neighbors(u)} (Pref(i,n) * Sim(n,u))}{\sum_{n \in Neighbors(u)} Sim(n,u)} \tag{2.2}$$

Where function *Pref(i,n)* denotes preference of user n for item i and function *Sim(n,u)* denotes similarity of users n and u which will be defined later.

We now need to define algorithm for finding most similar neighbors. One of very well known algorithms is called *Nearest N User Neighborhood* and is defined as follows:

Algorithm 2 Nearest N User Neighborhood

```
1: procedure FIND NEIGHBORS(theUser)
2:     users ← DATAMODEL.GETUSERS()
3:     topUsers ← EMPTY LIST
4:     for all user ← users do
5:         similarity ← USER SIMILARITY(user, theUser)
6:         topUsers ← topUsers + (user, similarity)
7:     end for
8:     topUsers ← SORT(topUsers)
9:     return TAKE USERS(topUsers, N)
10: end procedure
```

The algorithm uses model for calculating user similarity, which is not necessarily the same model as used in Algorithm 1. It returns the N most similar users from all the users, where the user similarity is defined arbitrarily. In Algorithm 2 the SORT works by comparing the similarities and TAKE USERS just returns N best results.

In mathematical notation the algorithm would be written as follows:

$$Neighbors(u) = \{v \in U | Sim(u,v) \geq max_{w \in U}(Sim(u,w) | |\{x \in U | Sim(x,u) \geq Sim(u,w)\}| = N)\} \tag{2.3}$$

The user similarity is one of the essential parts of user based collaborative filtering algorithms. Euclidean distance of users' ratings is defined as:

Algorithm 3 Euclidean Distance Similarity

```
1: procedure USER SIMILARITY(user1, user2)
2:     itemsRatings1 ← DATAMODEL.GETITEMSRATINGS(user1)
3:     itemsRatings2 ← DATAMODEL.GETITEMSRATINGS(user2)
4:     commonItems ← GET COMMON ITEMS(itemsRatings1, itemsRatings2)
5:     diffSquared ← 0
6:     for all item ← commonItems do
7:         x ← itemsRatings1[item]
8:         y ← itemsRatings2[item]
9:         diffSquared ← diffSquared + (x - y)^2
10:    end for
11:    n ← LENGTH OF(commonItems)
12:    return 1/(1 + sqrt(diffSquared / n))
13: end procedure
```

First we extract only the items that both users rated. We will denote the number of such items as n. Then for each such item we extract rating of both users and we calculate squared difference of these two numbers. We then sum the squared differences. Then we apply the square root to calculate the Euclidean distance. We could simply write $1/(1 + \sqrt{diffSquared})$ but this method has a drawback that if many ratings are present with only small differences, it behaves similarly to a situation with few ratings with bigger differences. We want to

eliminate this by calculating the similarity as $1/(1 + \sqrt{diffSquared\ /\ n})$, i.e. dividing the sum of squared differences by n. The reason behind it is that the more ratings there are, the bigger chance is that the squared difference will be bigger. However, if it is small, it should not behave in the same way as in the case with few ratings.

In mathematical notation the algorithm would be written as follows:

$$Sim(u,v) = \frac{1}{1 + \sqrt{\frac{\sum_{i \in I^{u,v}} (R_i^u - R_i^v)^2}{|I^{u,v}|}}} \tag{2.4}$$

Where $I^{u,v}$ is a set of items rated by both users u and v and R_i^u denotes rating of item i by user u.

2.3 General item based collaborative filtering

Algorithm 4 Item based Collaborative Filtering

```
1: procedure ESTIMATE(user, item)
2:     totalPreference ← 0.0
3:     totalSimilarity ← 0.0
4:     ratedItems ← DATAMODEL.GETRATEDBYUSER(user)
5:     for all ratedItem ← ratedItems do
6:         preference ← DATAMODEL.GETPREFERENCE(user, ratedItem)
7:         similarity ← ITEM SIMILARITY(item, ratedItem)
8:         totalPreference ← totalPreference + preference * similarity
9:         totalSimilarity ← totalSimilarity + similarity
10:    end for
11:    return totalPreference / totalSimilarity
12: end procedure
```

As in the user based collaborative filtering algorithm, the similarity is one of the essential parts of user based collaborative filtering algorithms. However, the similarity is now calculated between items, not users. First we extract all the items rated by the specified user from the data model. Then for each such item we extract a preference of specified user to this item from the data model and we calculate similarity between this item and specified item. By multiplying these two numbers we calculate weighted preference. In the end we divide weighted preference by total weight (sum of all *similarities*).

Mathematically the algorithm could be written as:

$$Ibcf(i,u) = \frac{\sum_{j \in I^u} (Pref(j,u) * Sim(i,j))}{\sum_{j \in I^u} Sim(i,j)} \tag{2.5}$$

where I^u is a set of all items rated by user u.

Similarly to the euclidean distance between two users, the euclidean distance between two items is defined as:

Algorithm 5 Euclidean Distance Similarity

```
1: procedure ITEM SIMILARITY(item1, item2)
2:     usersRatings1 ← DATAMODEL.GETUSERSRATINGS(item1)
3:     usersRatings2 ← DATAMODEL.GETUSERSRATINGS(item2)
4:     commonUsers ← GET COMMON USERS(usersRatings1, usersRatings2)
5:     diffSquared ← 0
6:     for all user ← commonUsers do
7:         x ← usersRatings1[user]
8:         y ← usersRatings2[user]
9:         diffSquared ← diffSquared + (x - y)^2
10:    end for
11:    n ← LENGTH OF(commonItems)
12:    return 1/(1 + sqrt(diffSquared / n))
13: end procedure
```

This algorithm is very similar to previously mentioned algorithm for calculating user similarity. The only difference is that in a user-item matrix with ratings we switched users and items, or put in other words, we transposed the matrix.

The mathematical notation for this algorithm looks like follows:

$$Sim(i,j) = \frac{1}{1 + \sqrt{\frac{\sum_{u \in U_{i,j}} (R_i^u - R_j^u)^2}{|U_{i,j}|}}} \tag{2.6}$$

where $U_{i,j}$ is a set of all users who rated both item i and item j.

2.4 Average models for recommendation

These models are based on calculating average rating, either average of all the user's ratings, or average of one item as rated by all the users. We will use:

1. User average
2. Item average

The *User average* model calculates the missing ratings as an average value of all previous ratings of user u:

$$\hat{r}_i^u = \frac{\sum_{j \in I^u} r_j^u}{|I^u|} \tag{2.7}$$

where I^u denotes all the items rated by user u.

The *Item average* model works similarly to *User average* model but the average is calculated from all the ratings of the given item i:

$$\hat{r}_i^u = \frac{\sum_{v \in U_i} r_i^v}{|U_i|} \tag{2.8}$$

where U_i denotes all the users who rated item i.

The *Item User Average* model is another variation of averages:

$$\hat{r}_i^u = \frac{\sum_{v \in U_i} r_i^v}{|U_i|} + \frac{\sum_{j \in I^u} r_j^u}{|I^u|} - \frac{\sum_{r_j^v \in R} r_j^v}{|R|} \tag{2.9}$$

where R denotes all the ratings in dataset.

We propose slightly modified model called *Item And User Average* which averages user and item averages:

$$\hat{r}_i^u = \frac{\sum_{v \in U_i} r_i^v}{2 * |U_i|} + \frac{\sum_{j \in I^u} r_j^u}{2 * |I^u|} \tag{2.10}$$

Each of the equations (2.7), (2.8), (2.9), (2.10) represents its own model for recommendation.

2.5 Hybrid algorithms

We propose a hybrid CF algorithm based on General user based collaborative filtering. The User Similarity model is based on individual attributes of an item. Let S_{A_i} denote a set of attribute values for attribute A_i. The attribute can be e.g. genres of a movie and the set of attribute values can be e.g. Drama and Thriller. Let r_v^u denote average rating of items that have attribute value v rated by user u. To calculate User Similarity of users u_1 and u_2 on one categorical attribute A_i we take all the common attribute values of a single attribute from items rated by both users and we calculate the normalized average of their differences:

$$sim_{A_i}(u_1, u_2) = 1 - \frac{\sum_{s \in S_{A_i}^{u_1} \bigcap S_{A_i}^{u_2}} |r_s^{u_1} - r_s^{u_2}|}{MaxDiff * |\{S_{A_i}^{u_1} \bigcap S_{A_i}^{u_2}\}|} \tag{2.11}$$

where *MaxDiff* denotes maximum of difference in rating, e.g. when the rating is on scale 1 to 5 then *MaxDiff* is equal to 4. It is easy to see that $sim_A \in [0, 1]$.

To calculate User Similarity of users u_1 and u_2 on one numerical attribute A_i we take the preferred values of both users for this attribute and we normalize the difference of these two values:

$$pref_{A_i}^u = \frac{\sum_{j \in I^u} r_j^u * v_j^{A_i}}{\sum_{j \in I^u} r_j^u} \tag{2.12}$$

where $v_j^{A_i}$ denotes the value of attribute A_i for item j. The User Similarity is then defined as:

$$sim_{A_i}(u_1, u_2) = \frac{|pref_{A_i}^{u_1} - pref_{A_i}^{u_2}|}{MaxDiff_{A_i}} \tag{2.13}$$

where $MaxDiff_{A_i}$ denotes maximum of differences of values for the attribute A_i.

Sometimes we use following model to calculate similarities of numerical attributes as follows:

$$sim_{A_i}(u_1, u_2) = \frac{|var_{A_i}^{u_1} - var_{A_i}^{u_2}|}{MaxVar} \tag{2.14}$$

where $var^u_{A_i}$ is the variance of the user's rating of items with attribute A_i and $MaxVar$ is the maximal possible variance in the ratings.

To calculate the similarity on a set of attributes A we are interested in, we define:

$$sim(u_1, u_2) = \frac{\sum_{A_i \in A} sim_{A_i}(u_1, u_2) * weight_{A_i}}{\sum_{A_i \in A} weight_{A_i}} \tag{2.15}$$

as the similarity of the users u_1 and u_2. The weight $weight_{A_i}$ for attribute A_i is used to tune the model in more details and in this thesis we put $weight_{A_i} = 1$.

Next, we propose a hybrid CF algorithm based on General item based collaborative filtering. The Item Similarity model is based on similar attributes of the two items. Let S_{A_k} denote a set of attribute values for attribute A_k. Let $S^i_{A_k}$ denote a set of attribute values for attribute A_k which the item i contains. To calculate Item Similarity of items i_1 and i_2 on one categorical attribute A_i we take all the common attribute values from the two items and we calculate the normalized similarity:

$$sim(i_1, i_2) = \frac{|\{a | a \in S^{i_1}_{A_k} \bigcap S^{i_2}_{A_k}\}|}{min(|S^{i_1}_{A_k}|, |S^{i_2}_{A_k}|)} \tag{2.16}$$

Another model we propose is also based on General user based collaborative filtering with idea taken from [1]. In this model the similarity of users u_1 and u_2 is calculated as similarity of variances of ratings. For nominal attributes we define similarity of users u_1 and u_2 as:

$$sim_{A_i}(u_1, u_2) = \frac{1}{1 + |\frac{(weight_{A_i}(u_1) - weight_{A_i}(u_2))}{MaxVar}|} \tag{2.17}$$

where $MaxVar$ is maximum variance of ratings. For example, when ratings are from interval $[0, 4]$, variance of ratings is maximum if they are minimum and maximum numbers on the rating scale, i.e. 0 and 4 in this case. The value of $MaxVar$ will be 4; and where

$$weight_{A_i}(u_i) = \frac{\sum_{s \in S_{A_i}} Var_s(u_i)}{|S_{A_i}|} \tag{2.18}$$

where $Var_s(u_i)$ is variance of ratings among all rated items by user u_i for single attribute value s. Variance is defined as follows:

$$Var(X) = E[(X - E[X])^2] \tag{2.19}$$

2.6 Content Based algorithms

Direct approach of calculating the user preference for item i led us to definition of the next model: first we fix one attribute A_k. Then for each attribute value from set of attribute values S_{A_k} we calculate average rating by the given user. Then we take only the attribute values of the item i that the user u rated and we calculate an average of the ratings for these attribute values. By calculating this for all the attributes we get the model:

$$r(u,i) = \frac{\sum_{v \in \cup_{k \in A} S_{A_k}} avg(v,u)}{|\cup_{k \in A} S_{A_k}|} \tag{2.20}$$

where $avg(v,u)$ denotes average rating of property value v by user u.

From the classical data mining algorithms we chose decision trees, Naive Bayes algorithm and logistic regression. Decision trees family of algorithms in this thesis consists of C4.5, Random Tree and Random Forest algorithms. We propose an algorithm which combines what we call global and local classifiers. The global classifier is trained on the whole dataset while the local classifier is trained for each user only on the items rated by the user. The result is a combination of the two results:

$$C(u,i) = \frac{C_{global}(u,i) * weight_{global} + C_{local}(u,i) * weight_{local}}{weight_{global} + weight_{local}} \tag{2.21}$$

where C is the result of combined classifier, C_{global} is the result of the global classifier and C_{local} is the result of the local classifier.

2.7 Matrix Factorization

These models work using only ratings and do not take any content into account. For experiments we used *SVD++ Factorization* as described in [3] and *Alternating-Least-Squares with Weighted-λ-Regularization* as described in [4]. These models were designed for famous *Netflix Prize* competition. In this thesis we do not try to alter these algorithms or enhance them in any way. Therefore we do not include more specific description of these algorithms.

3. Metrics

We use a few different metrics to measure how good the predictions of the models are.

3.1 Root mean square error

This metrics is one of the most frequently used metrics in various publications. It is defined as follows:

$$RMSE(\hat{r}) = \sqrt{E[(\hat{r} - r)^2]} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{r}_i - r_i)^2}{n}} \tag{3.1}$$

From the definition of *RMSE* is obvious that the penalty for incorrect estimate of rating grows quadratically with the difference of estimated rating and observed rating. On the other hand, it also obvious that the metric is symmetric, because the penalization is equal if the predicted rating and observed rating are 1 and 5 respectively and if they are 5 and 1 respectively. This might not always be desired. Therefore we propose similar metrics with slightly different characteristics.

3.2 Weighted root mean square error

We propose weighted *RMSE* which we expect to correct one of the characteristics of **RMSE** where predicted rating and observed rating with values 1 and 5 respectively and another situation with values 5 and 1 respectively yield the same penalty. We want to give higher penalty for wrong predictions $\hat{r}$ where the rating r is high and vice versa. The reason behind this proposal is that we are often interested in good prediction of high rated items and we do not give very much emphasis to the low rated items.

$$RMSE(\hat{r}) = \sqrt{\frac{\sum_{i=1}^{n}((\hat{r}_i - r_i) * weight_{r_i})^2}{n}} \tag{3.2}$$

where $weight_r$ is calculated as follows:

$$weight_r = \frac{r}{maxRating} \tag{3.3}$$

so that on scale 1 to 10 the $weight_r$ has values 1/10 to 1. If the scale is something different, i.e. 0 to 4 as it is in the *Sushi* dataset, we correct the ratings, so that $weight_r$ has values 1/5 to 1, as there are 5 different ratings.

3.3 Kendall's tau

This metrics is a measure of similarity of two orderings or in other words, a similarity of rank correlation. Let us have two orderings $u_1 \dots u_n$ and $v_1 \dots v_n$ of the same numbers, such that v_i and u_i are unique. Then we say the pair (u_i, v_i) and (u_j, v_j) is concordant if $u_i < u_j$ and $v_i < v_j$ or $u_i > u_j$ and $v_i > v_j$. Otherwise this pair is discordant. Then we define Kendall's tau as:

$$\tau = \frac{\sum_{i,j>i,(u_i,v_i)and(u_j,v_j)concordant} 1 - \sum_{i,j>i,(u_i,v_i)and(u_j,v_j)discordant} 1}{1/2n * (n-1)} \tag{3.4}$$

Because there are exactly $1/2n*(n-1)$ pairs (u_i, v_i) and (u_j, v_j), the Kendall's tau is always in interval $[-1, 1]$.

4. Datasets

4.1 MovieLens dataset

The MovieLens datasets are well known and used datasets for experimentation in recommender systems research area. There are three sets, MovieLens 100k, MovieLens 1M and MovieLens 10M with approximately 100,000, 1,000,000 and 10,000,000 ratings respectively. In this thesis we use the MovieLens 1M dataset. The dataset contains preferences of users to a collection of movies. Ratings are on scale 1-5. We provide basic statistics about the set in the table 4.1.

Table 4.1: MovieLens1M statistics

Item count	3,706
User count	6,040
Rating count	1,000,209
Density	4.47%
Average rating count per item	269.89
Average rating count per user	165.60

4.2 Joined MovieLens and IMDb datasets

Along with ratings the dataset contains the title of each movie. The titles were used to gain more information about the movies by joining the dataset with information provided by IMDb. We were able to successfully add information about genres, directors, actors and actresses to the MovieLens1M movies information. The table 4.2 shows number of cases in which various attributes were added to the movies. In total there were 389 movies in which one of the four attributes could not be determined from the IMDb dataset. This is due to the fact that not every movie from MovieLens1M dataset exists in IMDb dataset. Also not every movie necessarily has actors or actresses, therefore we do not limit the dataset to only movies which were successfully matched in all four attributes. The number of movies in MovieLens1M dataset is 3,883, but only 3,706 of them are actually rated.

Table 4.2: Joining MovieLens1M and IMDb datasets

Attribute	Joined	Not joined
Genres	3,556	327
Directors	3,558	325
Actors	3,544	339
Actresses	3,514	369

4.3 Sushi dataset

This dataset contains preferences of sushi eaters to a collection of sushis. Each user rated exactly 10 pieces of sushi on scale 0-4. We provide basic statistics about the set in the table 4.3.

Table 4.3: Sushi statistics

Item count	100
User count	5,000
Rating count	50,000
Density	10%
Average rating count per item	500
Exact rating count per user	10

4.4 Notebooks dataset

The Notebooks dataset contains artificially generated preferences of users to notebook computers. Each user's ratings are based on a psychological model of preferences towards various parameters of notebooks. We provide basic statistics about the set in the table 4.4

Table 4.4: Notebooks statistics

Item count	200
User count	108
Rating count	21,600

5. Experiments

There are several models that were used using the same data to compare the models. Except of the proposed models based on the attributes of items we also tested slightly modified model called *User Based Collaborative Filtering* as implemented in *Taste* framework, version *0.8* using the distance measures *Euclidean distance* and *Pearson Correlation*. The modification was made to the class *GenericUserBasedRecommender* so that the estimated rating for an item is computed as an average of ratings of only those users that rated the item, not all the closest users. This modification increases the ratio of items that could be recommended to items that could not be recommended. In original implementation there could be situations where out of N closest users to the user u none of them rated the item i and therefore the average could not be computed.

Another model from *Taste* framework that we use is called *Item Based Collaborative Filtering* using the same distance measures as in previous case. *Taste* offers also so called *User Average* model as well as *Item Average* model.

The matrix factorization techniques offered by *Taste* framework are *SVD++ Factorizer* and *ALSWR Factorizer*, which stands for *Alternating-Least-Squares with Weighted- λ -Regularization*.

We abbreviate the algorithm names as follows:

- UBED - User Based Collaborative Filtering with Euclidean Distance
- UBPC - User Based Collaborative Filtering with Pearson Correlation Distance
- IBED - Item Based Collaborative Filtering with Euclidean Distance
- UA - User Average
- IA - Item Average
- IUA - Item User Average
- IAUA - Item And User Average
- SVDSGDxInF - Matrix Factorization using SVD++ Factorizer with x iterations and n features
- SVDALSWRxInF - Matrix Factorization using ALSWR Factorizer with x iterations and n features
- SO - SlopeOne algorithm

The calculations were run using 0.3 as evaluation percentage and 0.7 as training percentage: from the whole dataset there was randomly selected a set of users for evaluation and the rating from these users were randomly divided into training and testing set. Each experiment was executed 100 times.

The following table provides an overview of combination of datasets, metrics and models used in experiments:

Dataset	Model	Metrics
Movielens1M	UBED, UBPC, UBMLUS SVDSGD, SVDALSWR IBED, IBPC, UA, IA, IUA IAUA, IBMIS, SO, MLCB	RMSE WRMSE Tau
Sushi	UBED, UBS, UBW, UBWD SVDSGD, SVDALSWR IBED, IBPC, UA, IA, IUA IAUA, IBSIS, SO J48, RandomTree, Logistic, NaiveBayes, RandomForest	RMSE WRMSE Tau
Notebooks	UBED, UBPC, UBShdpmr SVDSGD, SVDALSWR IBED, IBPC, UA, IA IUA, IAUA, NTB, SO	RMSE WRMSE Tau

5.1 MovieLens1M experiments

For *MovieLens1M* dataset we used several models to make experiments. First family of experiments were *User Based* algorithms. The common thing in all the *User Based* algorithms we use is, that all have a certain number of closest neighbors for user u to compute the average rating. We denote this number by N. Then we used *Average* models and *Matrix Factorization*.

For *User Based* algorithm we use the following abbreviations:

- UBSMLUS - User Based CF with Movielens user similarity, i.e. similarities between two users are based on similarities of rating of common attributes: genres, directors, actors, actresses. This algorithm is described in section 2.5 in models 2.11 and 2.15.
- IBMIS - Item Based CF with Movielens item similarity, i.e. similarities between two items are based on the similarities of the genres, directors, actors and actresses of these two items The algorithm is described in section 2.5 in model 2.16.
- MLCB - Movielens Content Based recommender, where preference is calculated directly from user's preferences for the item's attributes. This algorithm can be found in section 2.6 in model 2.20

The following experiments were run using $RMSE$, $WRMSE$ and $Kendall'sTau$ metrics.

It is essential to notice that hybrid User Based algorithms did not perform worse than traditional User Based Algorithms. However their performance is not significantly better. Also note that Content Based algorithms in *Other Algorithms* section were neither worst, nor best of the algorithms plotted on the same graph.

Figure 5.1: Movielens RMSE User Based Algorithms

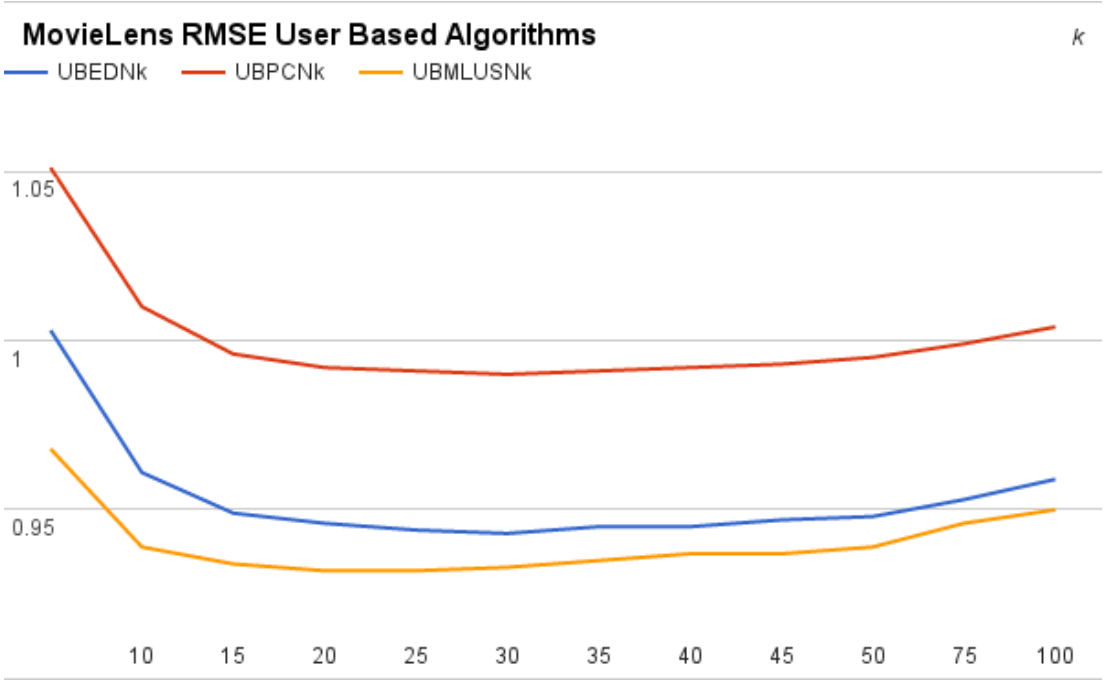


Figure 5.2: Movielens WRMSE User Based Algorithms

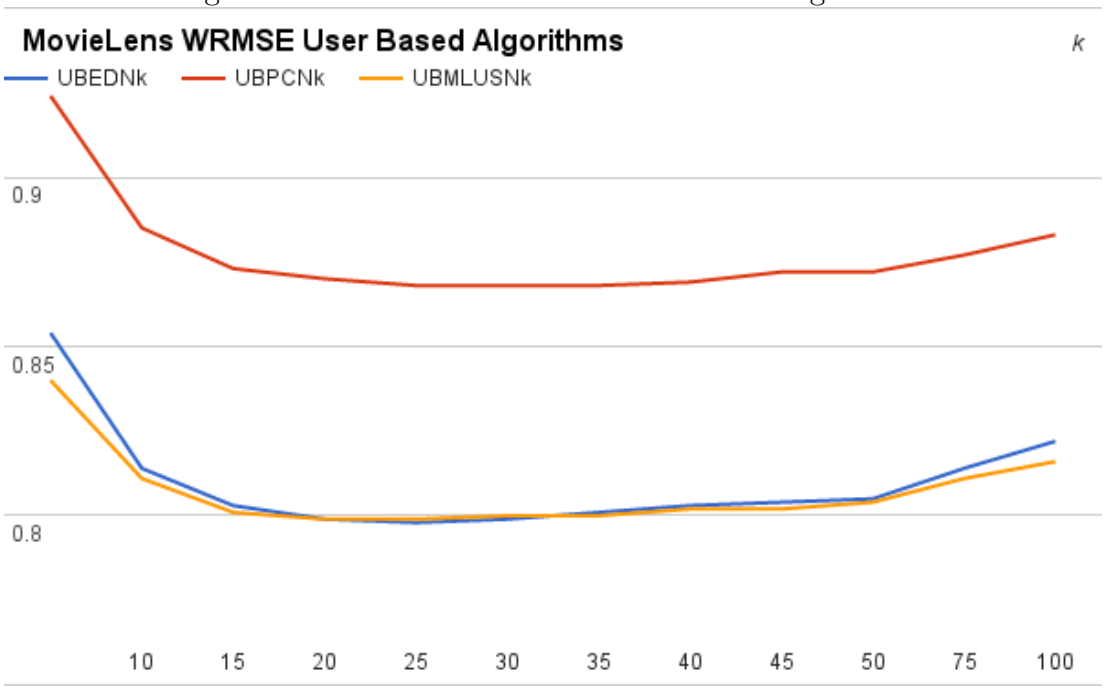


Figure 5.3: Movielens Tau User Based Algorithms

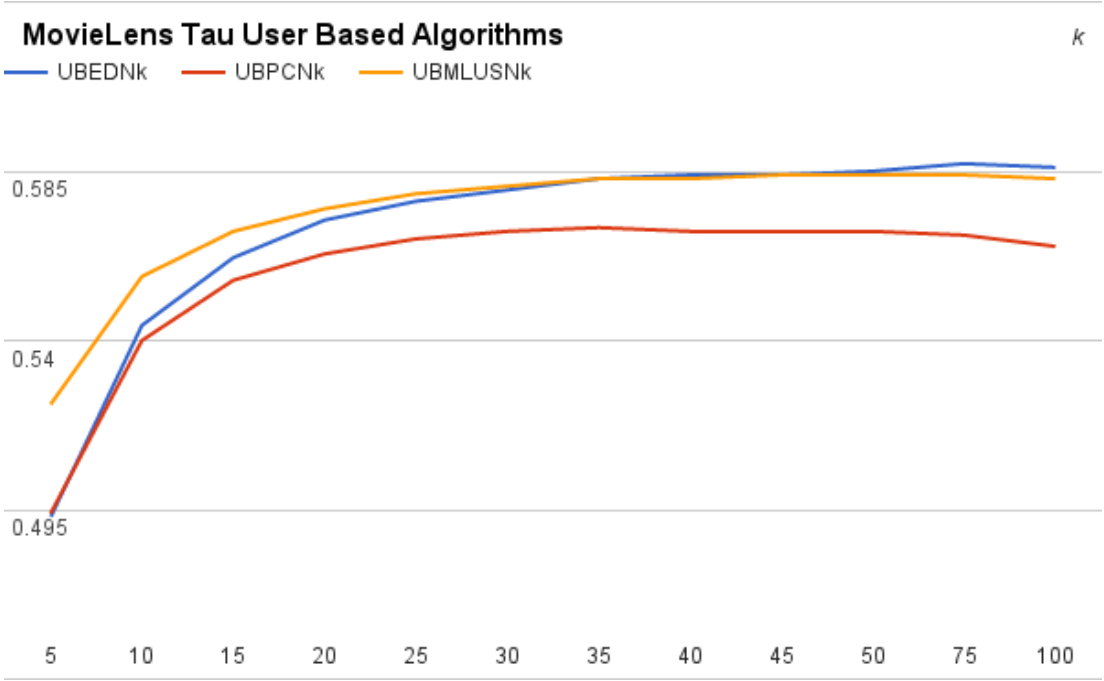


Figure 5.4: Movielens RMSE Factorization Algorithms

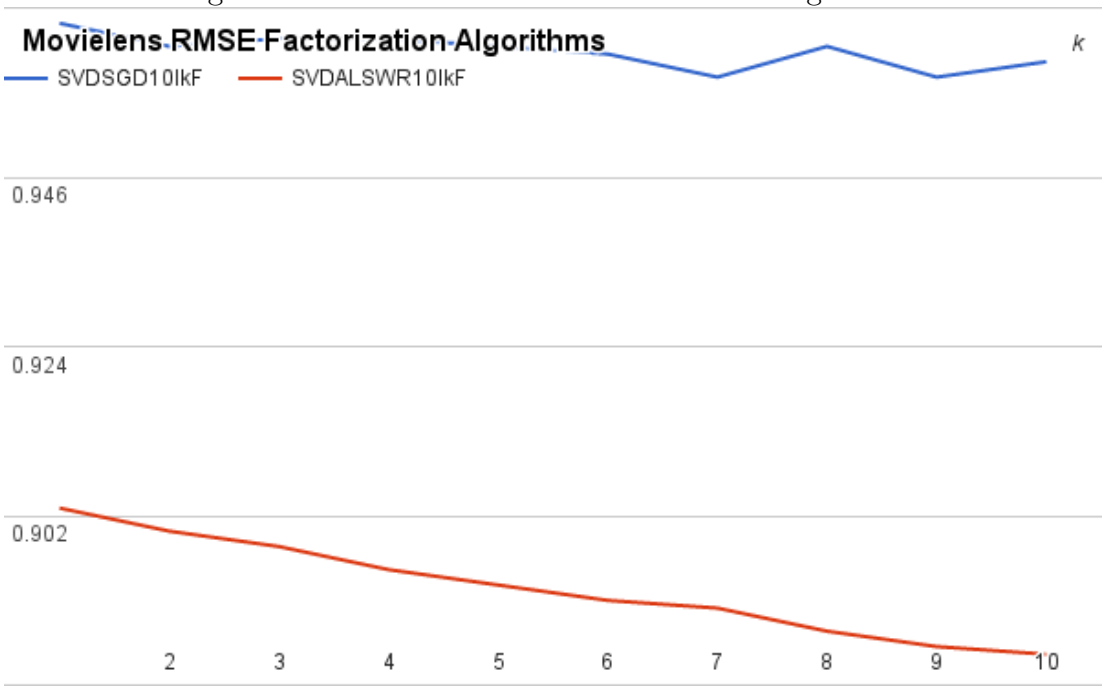


Figure 5.5: Movielens WRMSE Factorization Algorithms

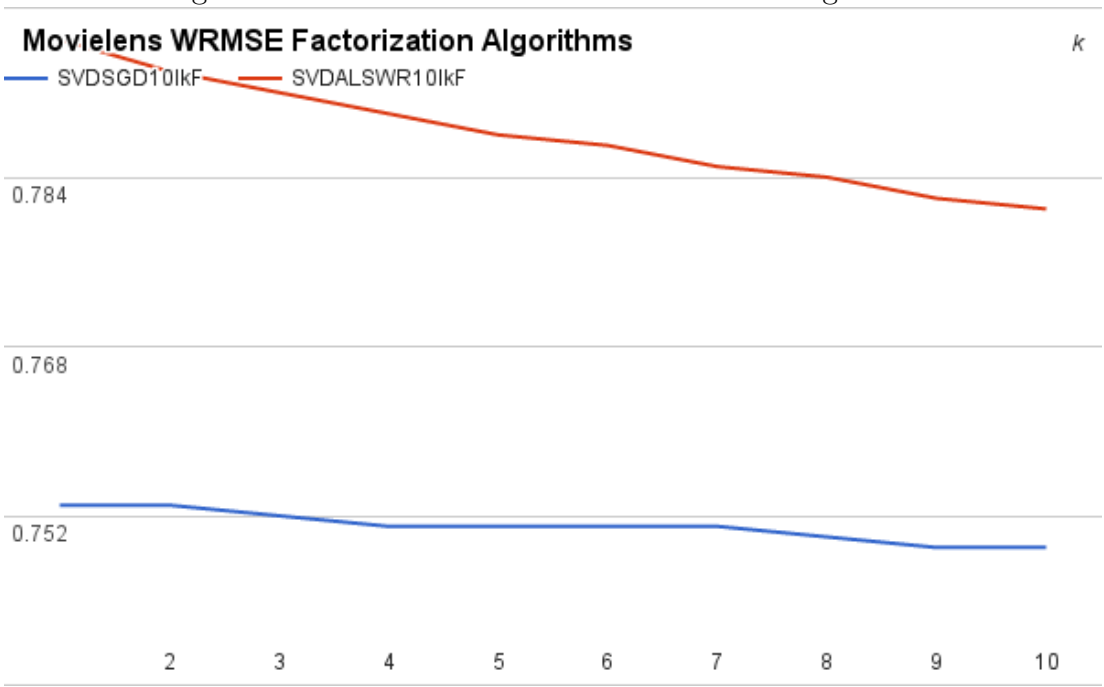


Figure 5.6: Movielens Tau Factorization Algorithms

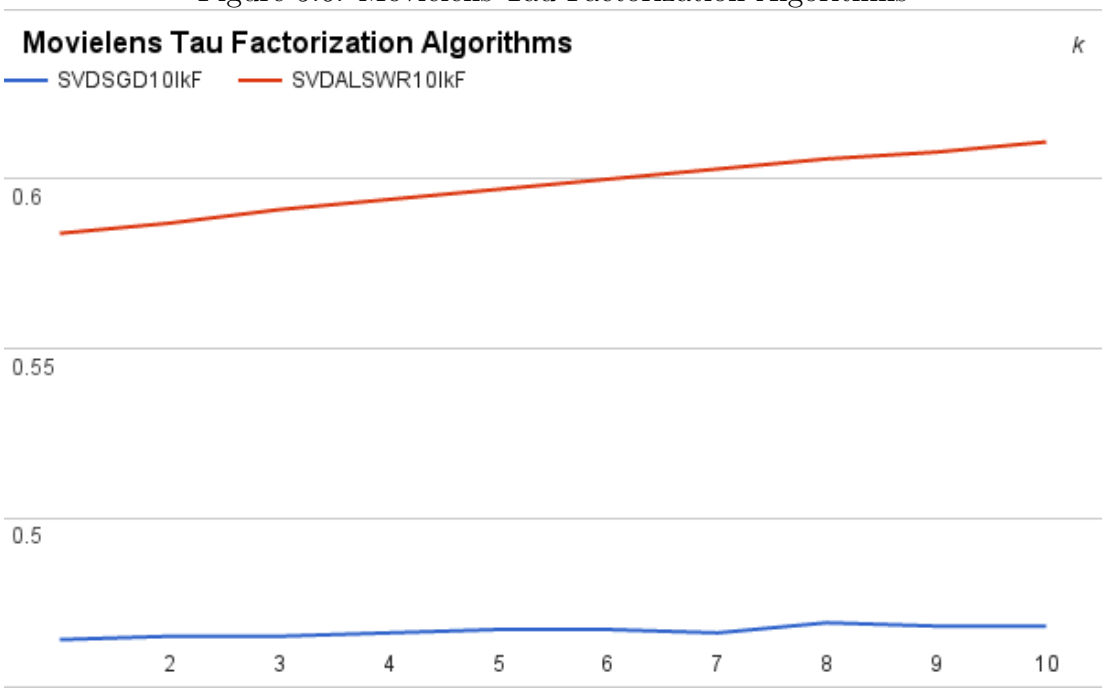


Figure 5.7: Movielens RMSE Other Algorithms

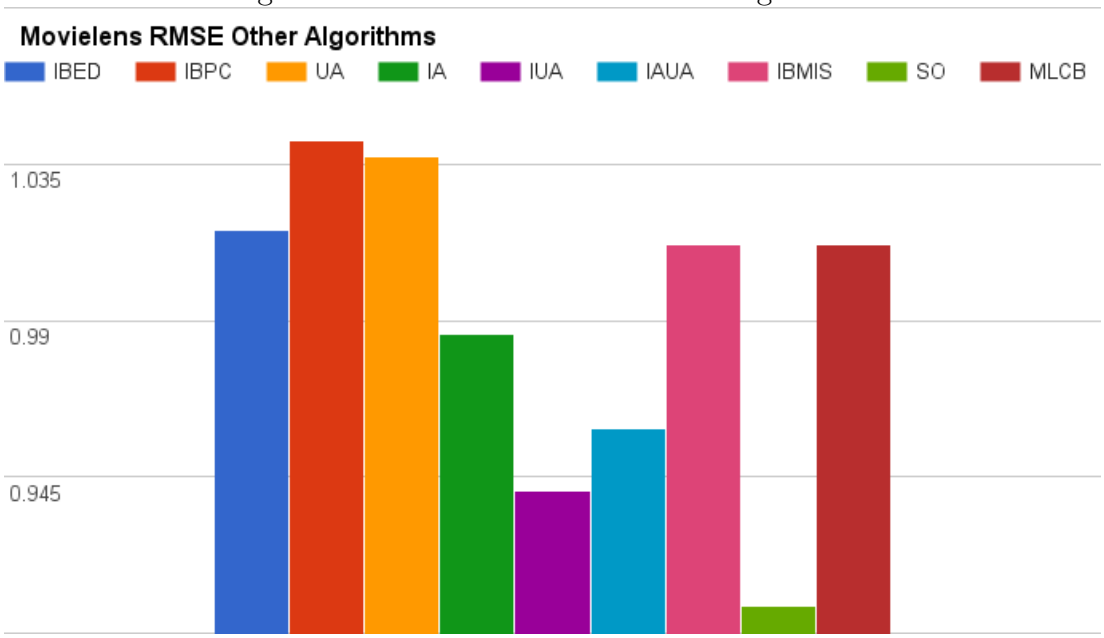


Figure 5.8: Movielens WRMSE Other Algorithms

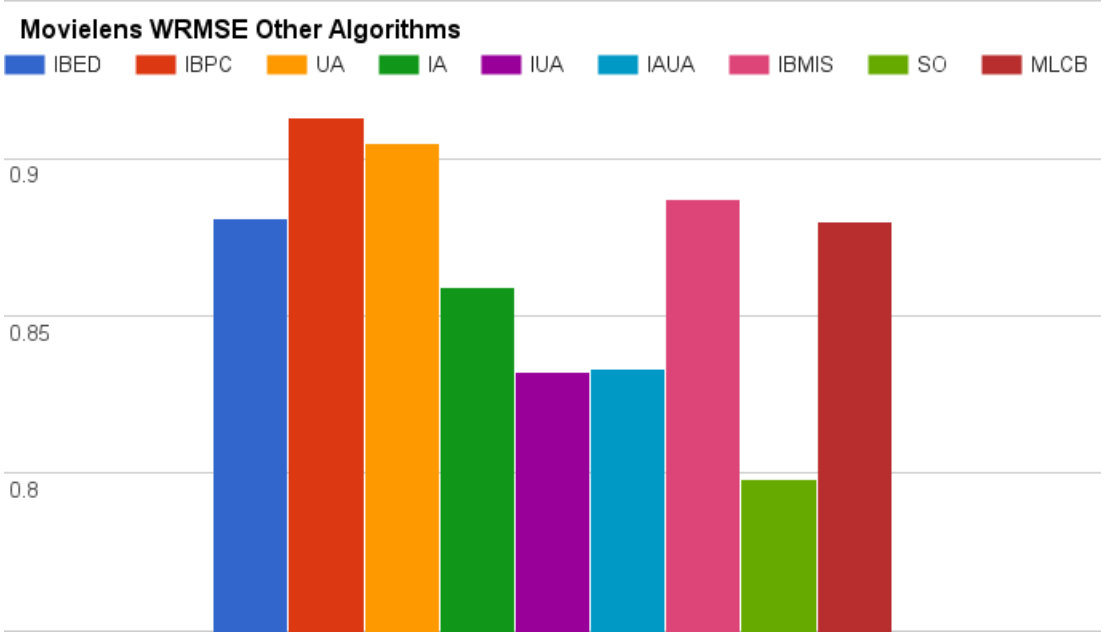
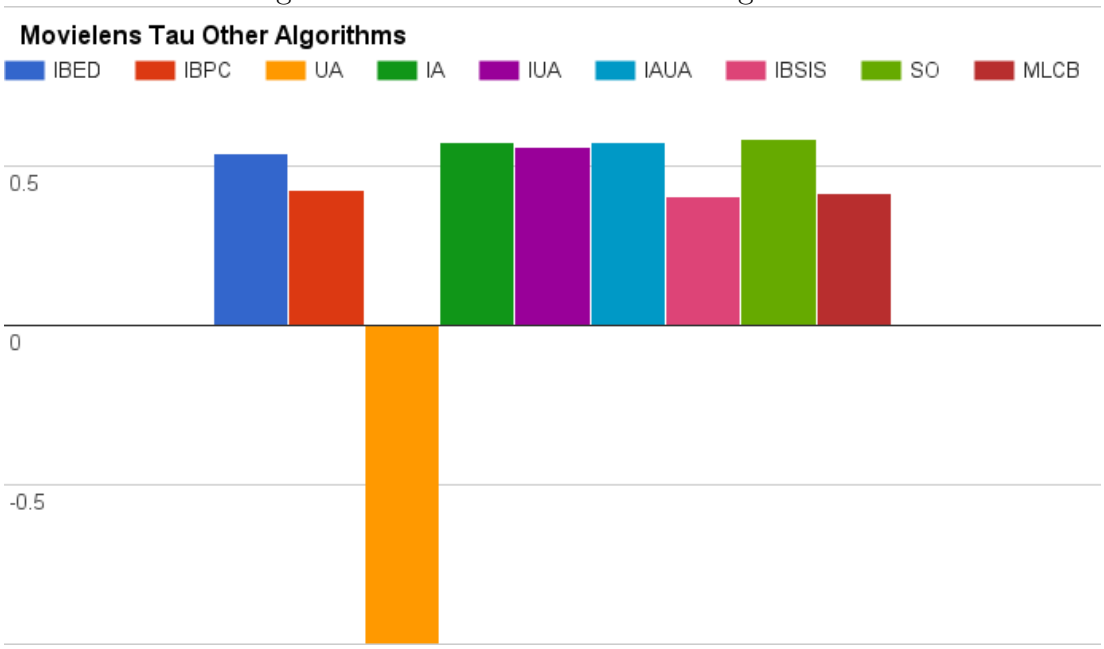


Figure 5.9: Movielens Tau Other Algorithms



5.2 Sushi experiments

For *Sushi* dataset we used the same models as for *MovieLens1M* dataset. For *User Based* algorithm we use the following abbreviations:

- UBSNk - User Based CF with distance of every attribute of both items and users, k nearest neighbors. It was executed on attributes major group, minor group, oiliness, price and style. This algorithm can be found in section 2.5 in models 2.11, 2.15 and 2.13.
- UBWNk - User Based CF with distance of weights of every attribute of both items and users, k nearest neighbors. It was executed on attributes major group, minor group, oiliness and price. This algorithm can be found in section 2.5 in models 2.17.
- UBWDNk - User Based CF with weighted distance of every attribute of both items and users, k nearest neighbors. It was executed on attributes major group, minor group, oiliness and price. This algorithm is described in section 2.5 in models 2.11, 2.15 and 2.14.
- IBSIS - Item Based CF with Sushi item similarity, i.e. similarities between two items are based on the similarities of the major group, minor group, oiliness, price and style of these two items The algorithm is described in section 2.5 in model 2.16.

We also used machine learning algorithms described in section Content Based algorithms. We abbreviate the models as follows:

- G+L - Global and Local
- G - Global
- L - Local
- G+L+U - Global and Local with User features

The following experiments were run using $RMSE$, $WRMSE$ and $Kendall'sTau$ metrics.

For Sushi dataset the *UBS* and *UBWD* models performed better than other User Based models. The difference is significant. Concerning the classifiers, we see that Random Forest algorithm performed best with combined Global and Local classifiers. Adding user data to the classifier did not yield any improvement. We also see that *IBSIS*, the hybrid Item Based algorithm did not perform better than *Average models*.

Figure 5.10: Sushi RMSE User Based Algorithms

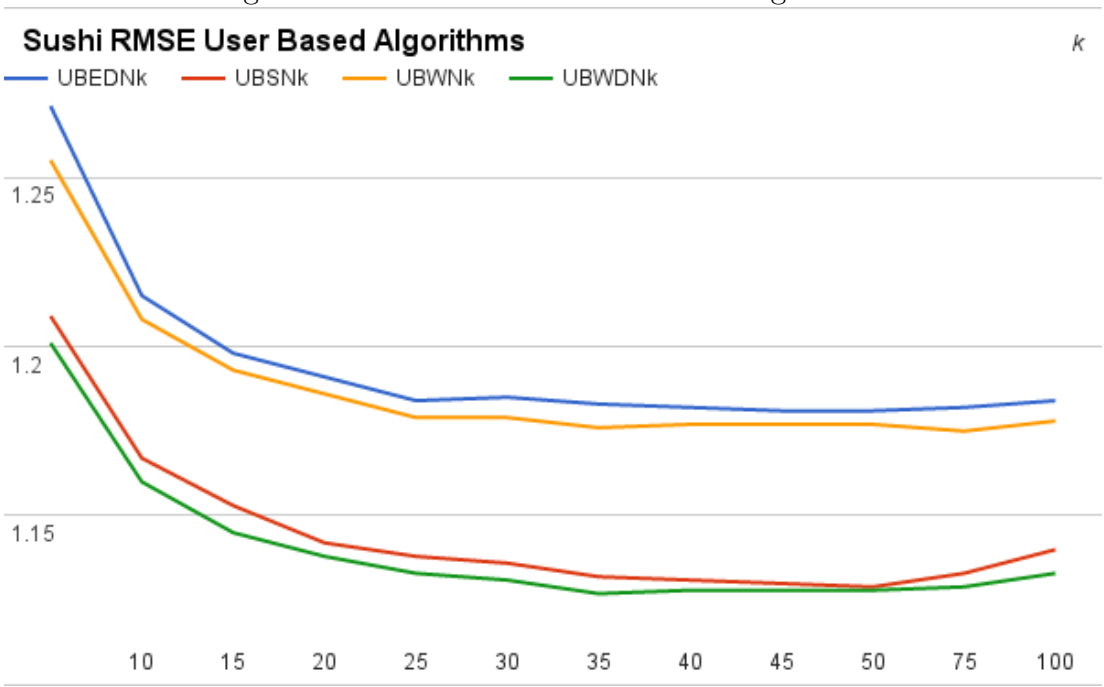


Figure 5.11: Sushi WRMSE User Based Algorithms

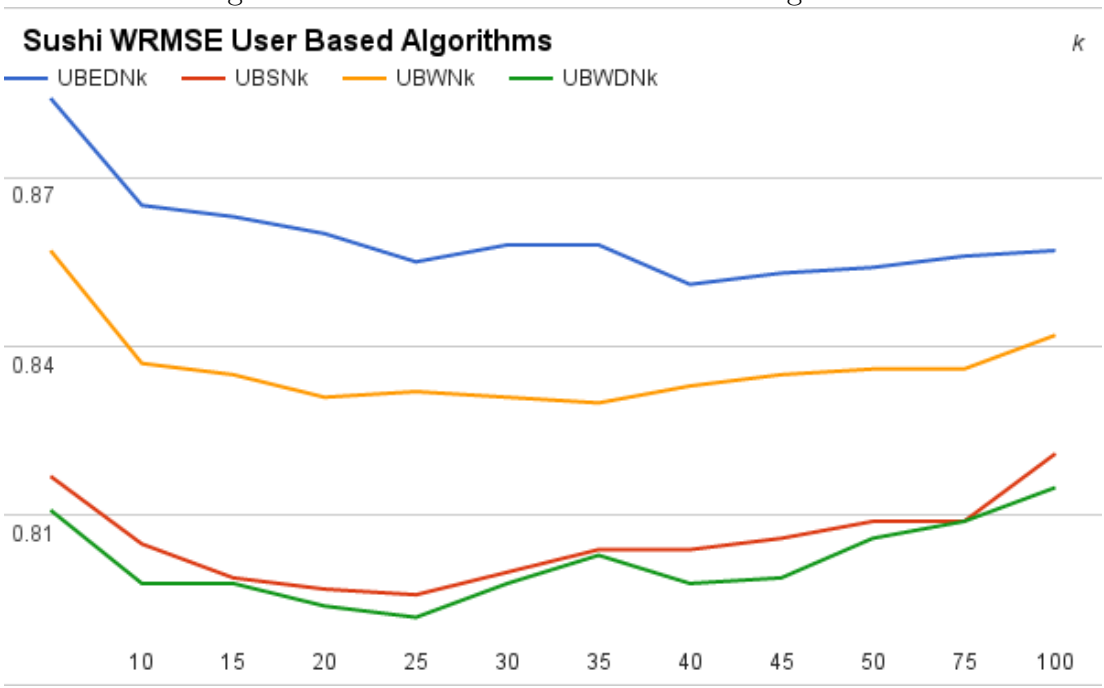


Figure 5.12: Sushi Tau User Based Algorithms

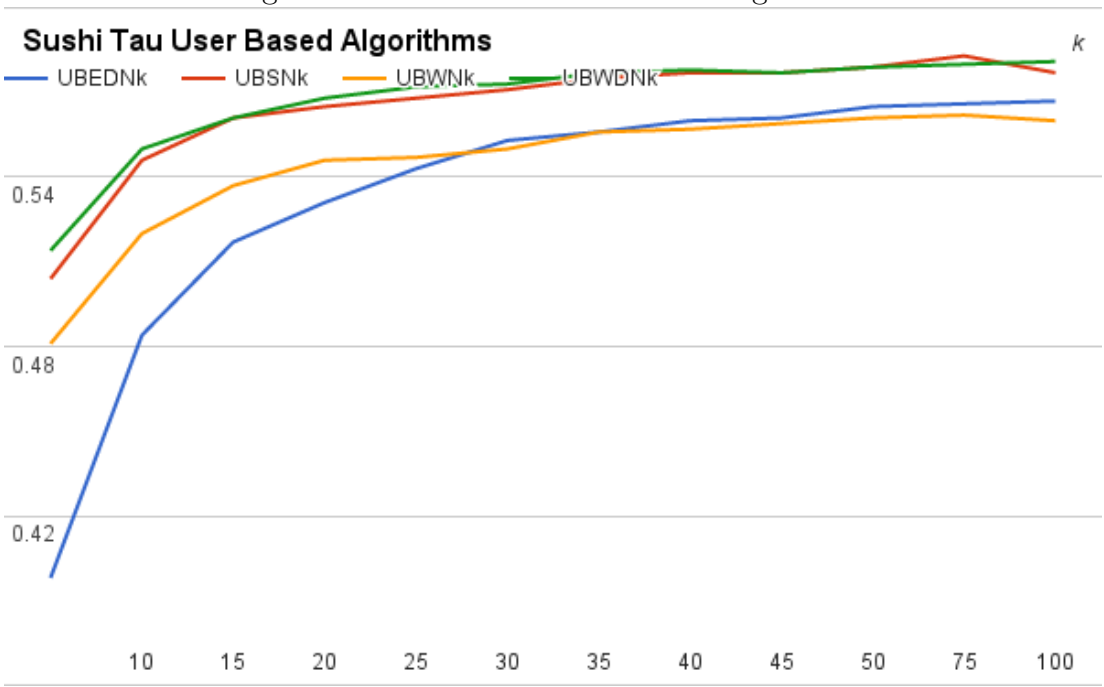


Figure 5.13: Sushi RMSE Factorization Algorithms

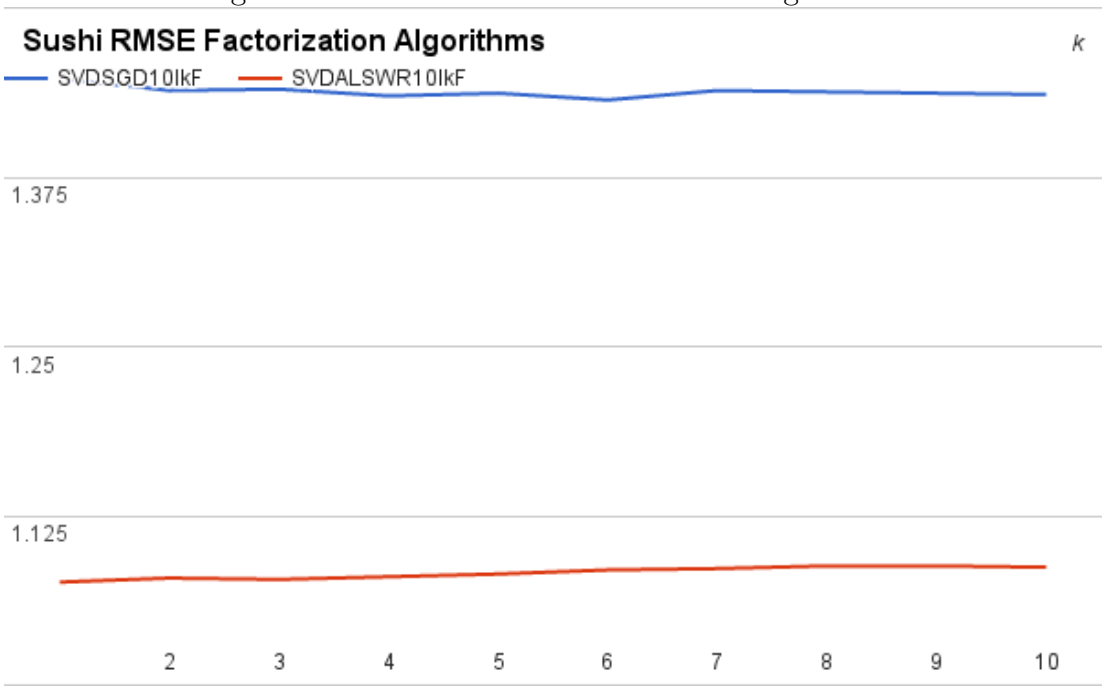


Figure 5.14: Sushi WRMSE Factorization Algorithms

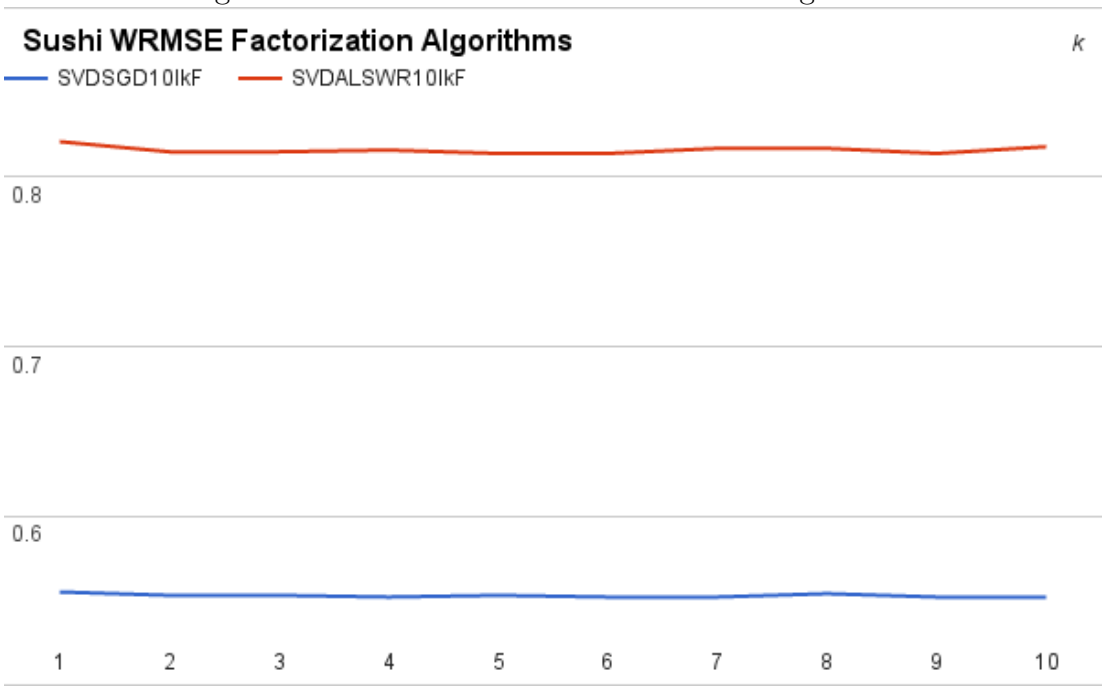


Figure 5.15: Sushi Tau Factorization Algorithms

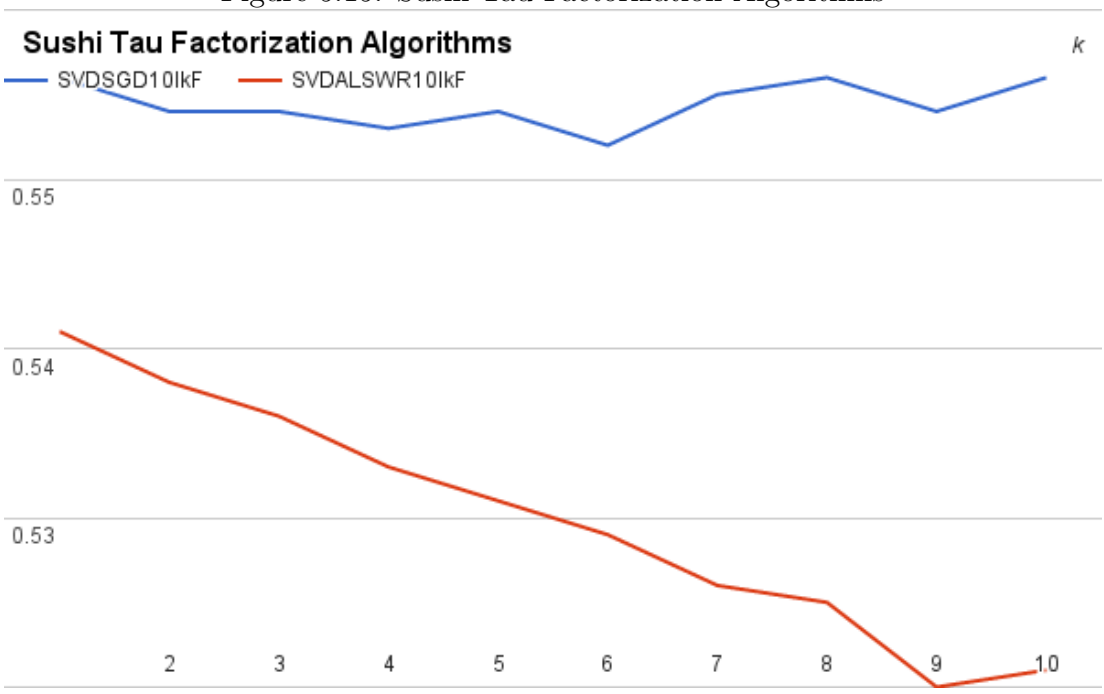


Figure 5.16: Sushi RMSE Classifiers

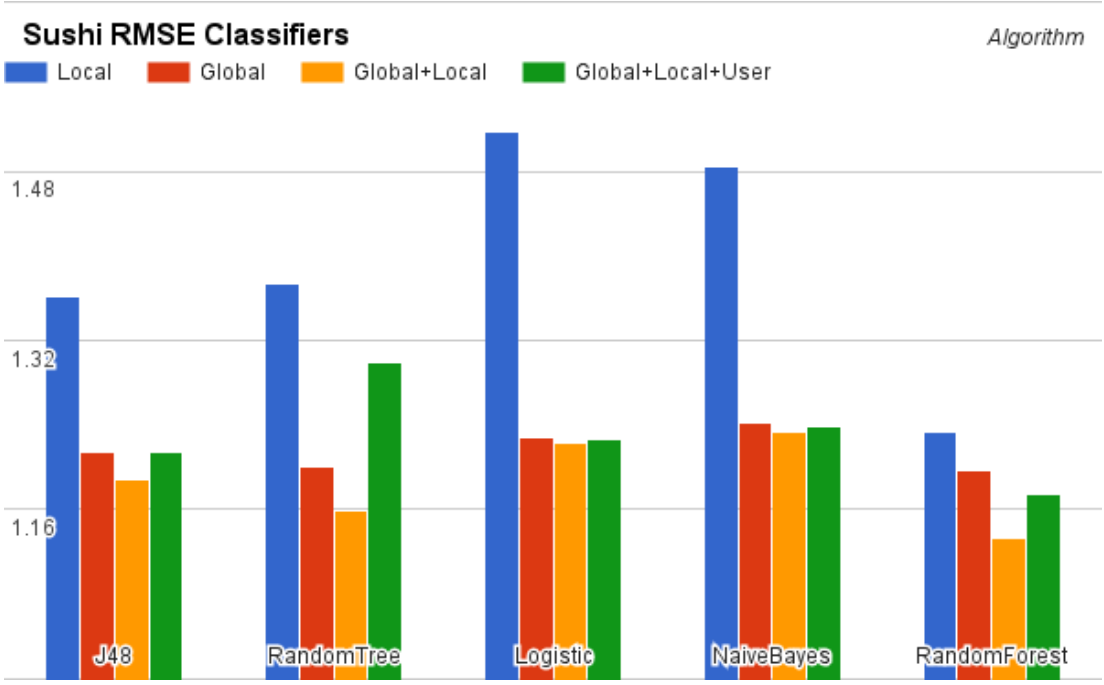


Figure 5.17: Sushi WRMSE Classifiers

Figure 5.18: Sushi Tau Classifiers

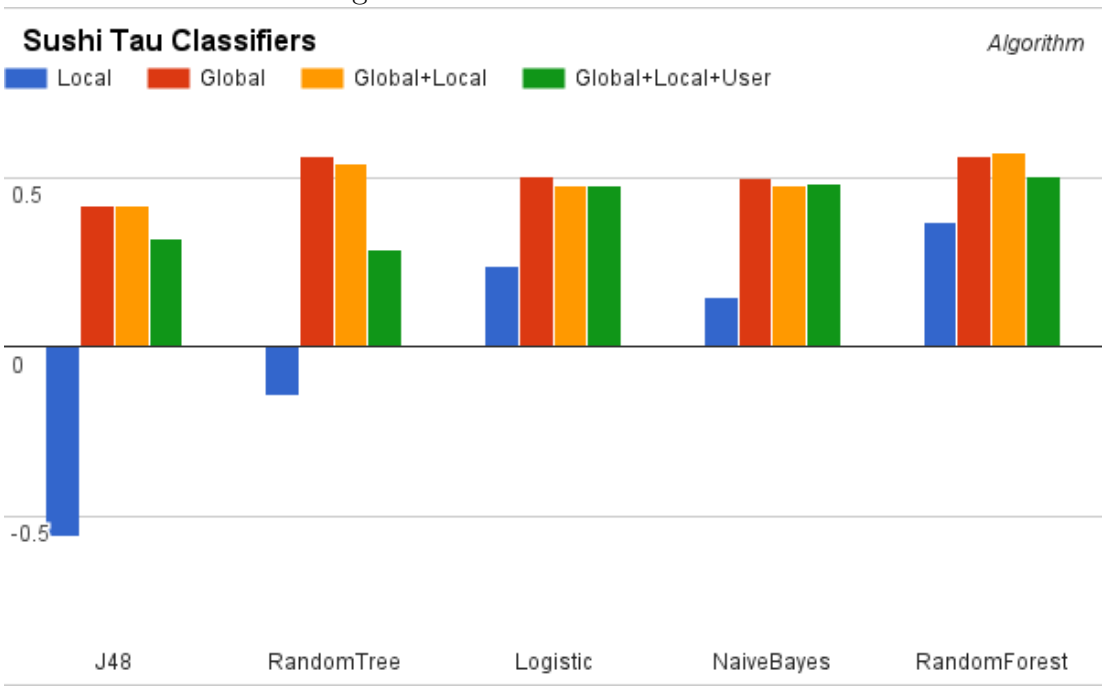


Figure 5.19: Sushi RMSE Other Algorithms

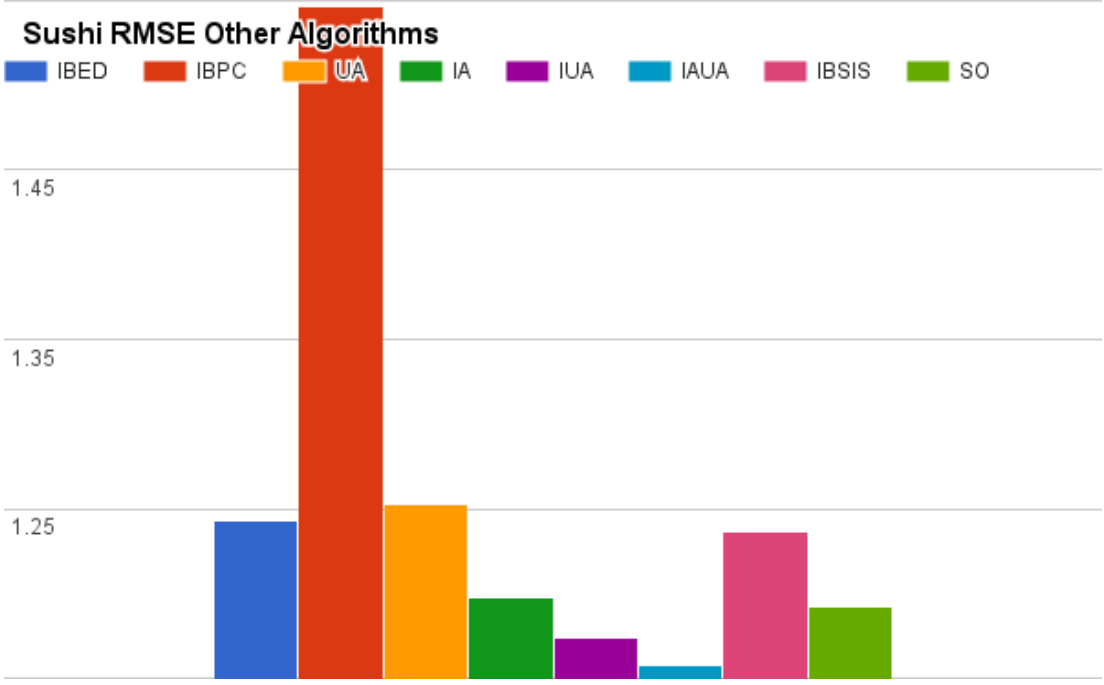


Figure 5.20: Sushi WRMSE Other Algorithms

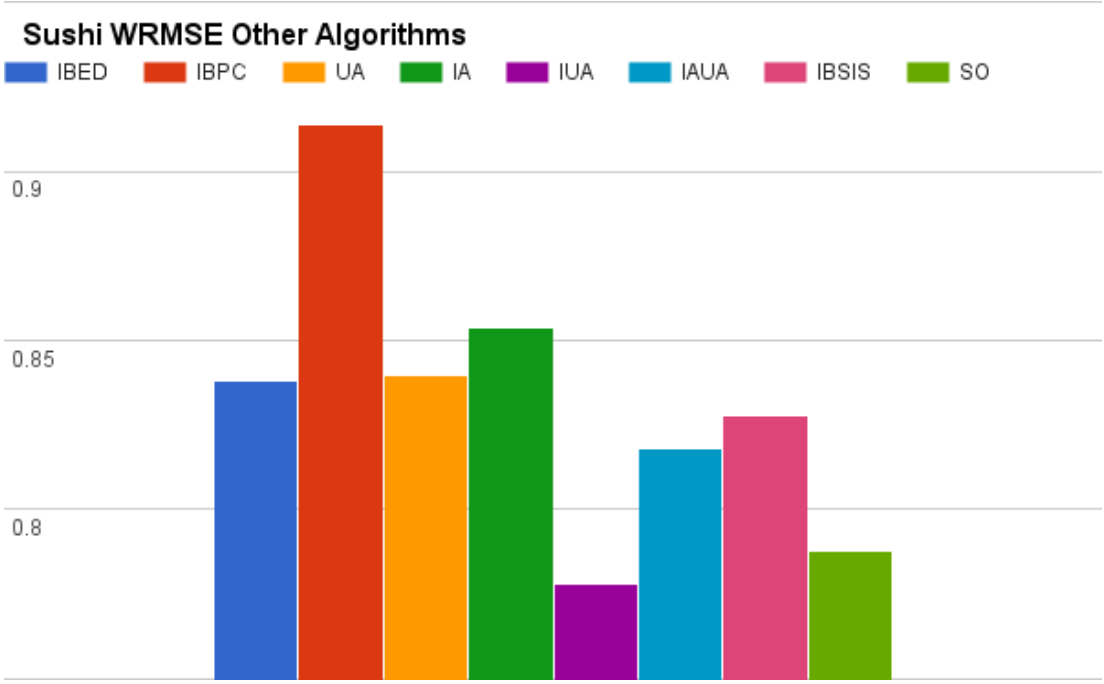
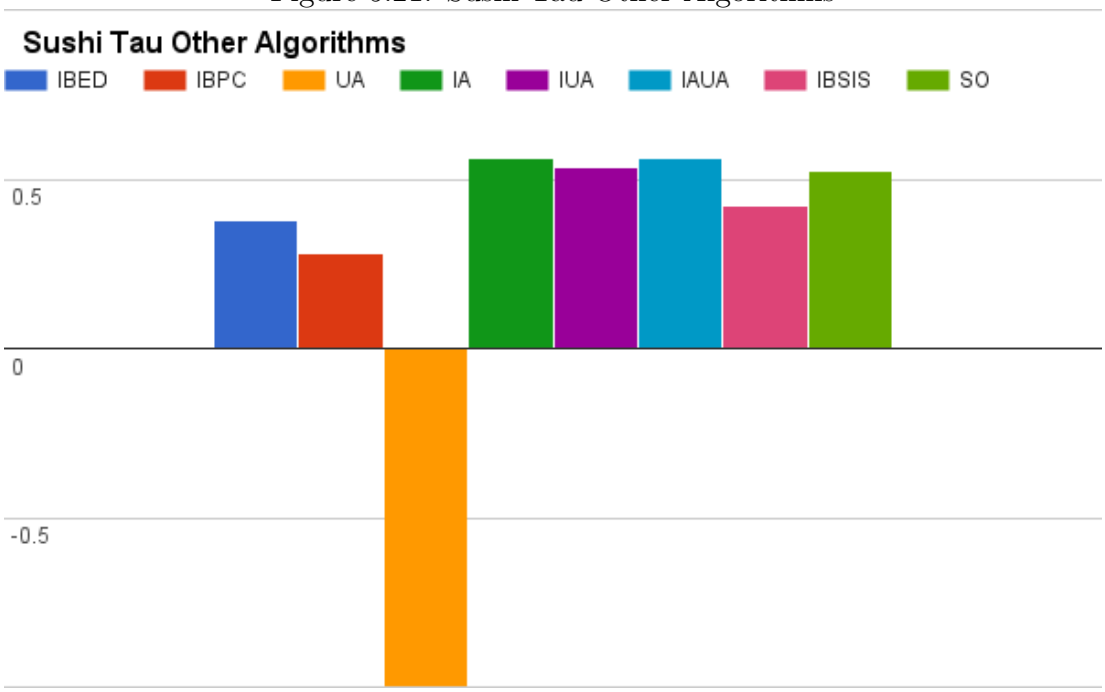


Figure 5.21: Sushi Tau Other Algorithms



5.3 Notebooks experiments

For *User Based* algorithm we use the following abbreviations:

- UBShdpmrNk - User Based CF with distance of HDD, display, price, manufacturer and RAM ratings with k neighbors. This algorithm can be found in section 2.5 in models 2.11, 2.15 and 2.13.
- NTB - Notebooks Content Based recommender, where preference is calculated directly from user's preferences for the item's attributes. This algorithm can be found in section 2.6 in model 2.20

The following experiments were run using $RMSE$, $WRMSE$ and $Kendall'sTau$ metrics.

For the Notebooks dataset we see that User Based hybrid algorithm did not perform better than standard User Based model with Euclidean Distance. The *NTB* model, based on content, was one of the worst models tested in Notebooks dataset.

Figure 5.22: Notebooks RMSE User Based Algorithms

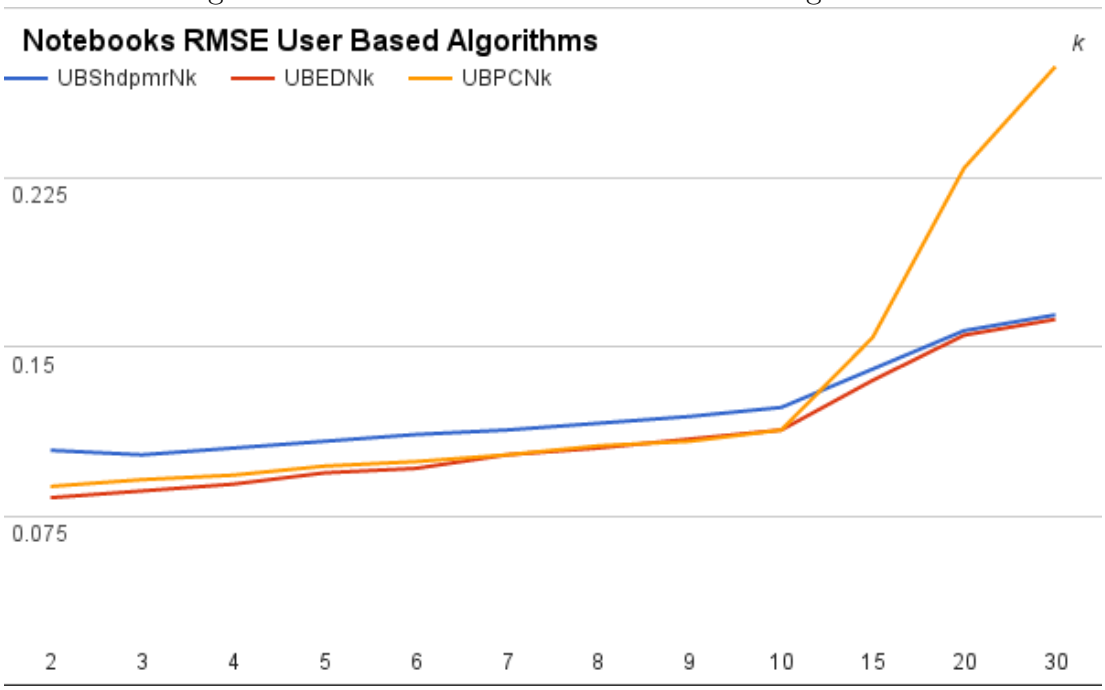


Figure 5.23: Notebooks WRMSE User Based Algorithms

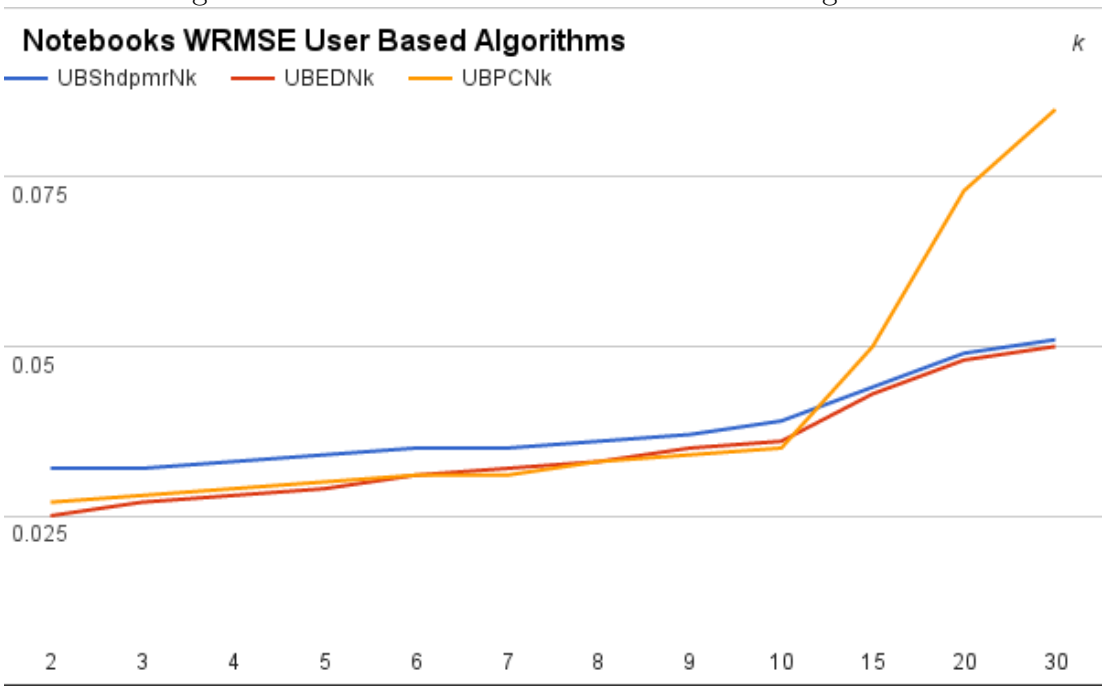


Figure 5.24: Notebooks Tau User Based Algorithms

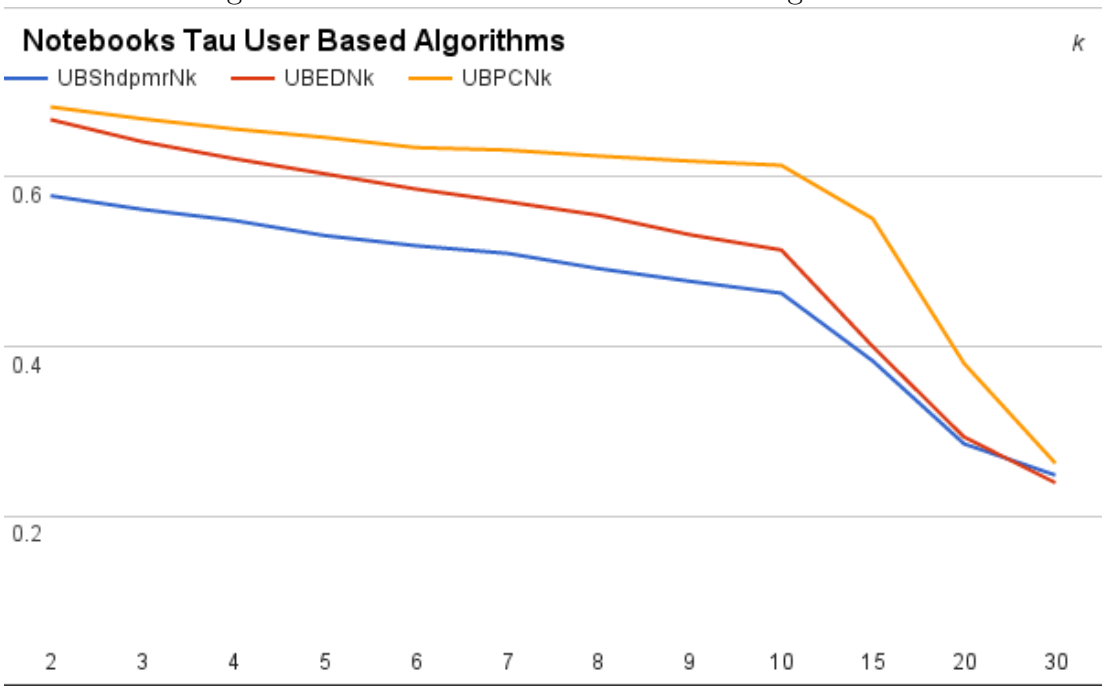


Figure 5.25: Notebooks RMSE Factorization Algorithms

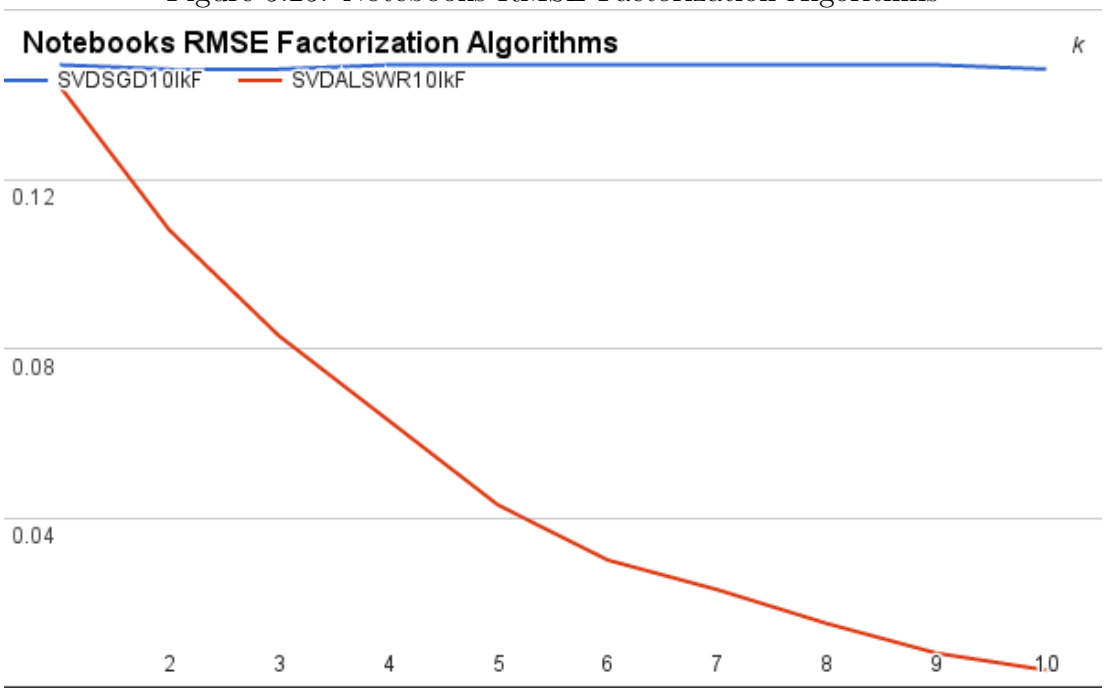


Figure 5.26: Notebooks WRMSE Factorization Algorithms

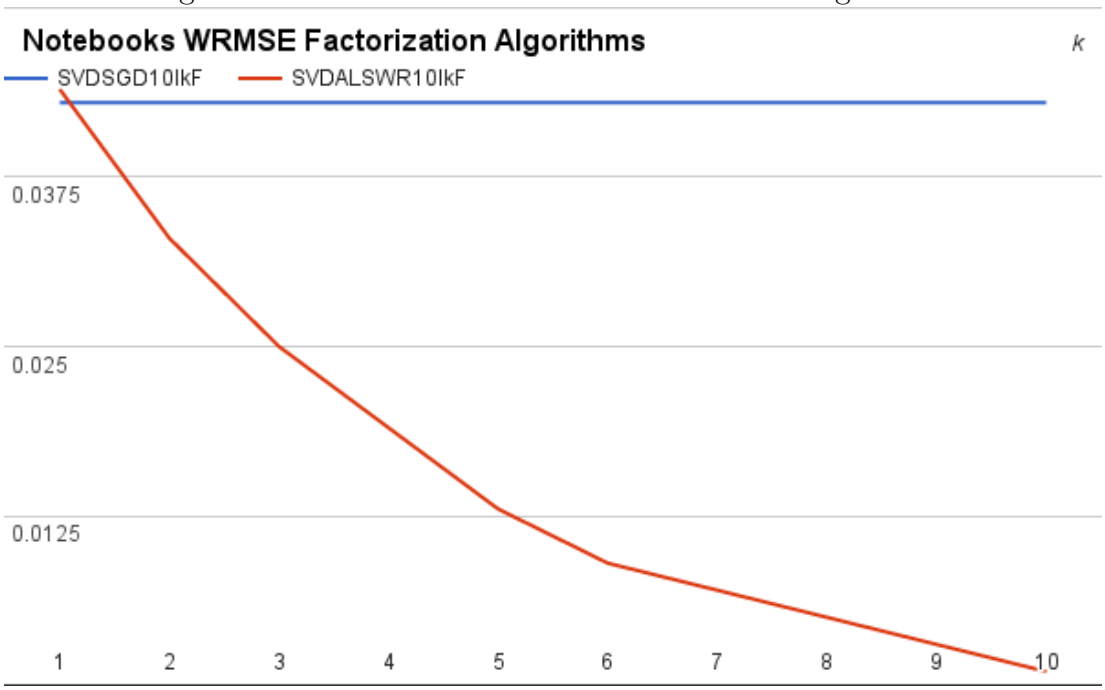


Figure 5.27: Notebooks Tau Factorization Algorithms

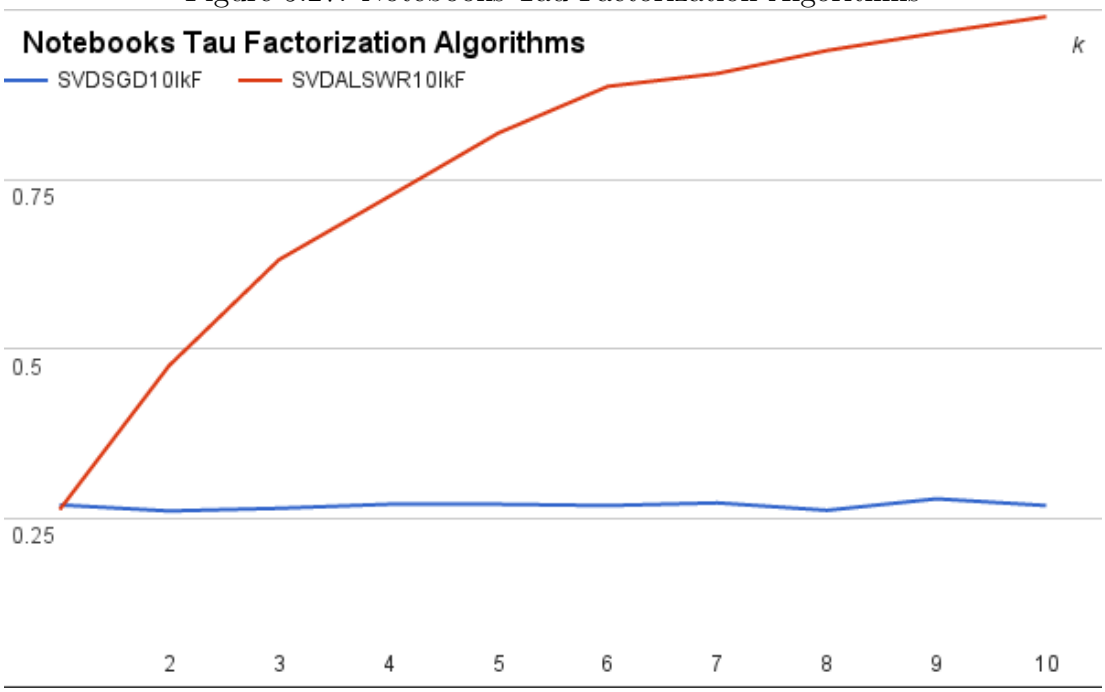


Figure 5.28: Notebooks RMSE Other Algorithms

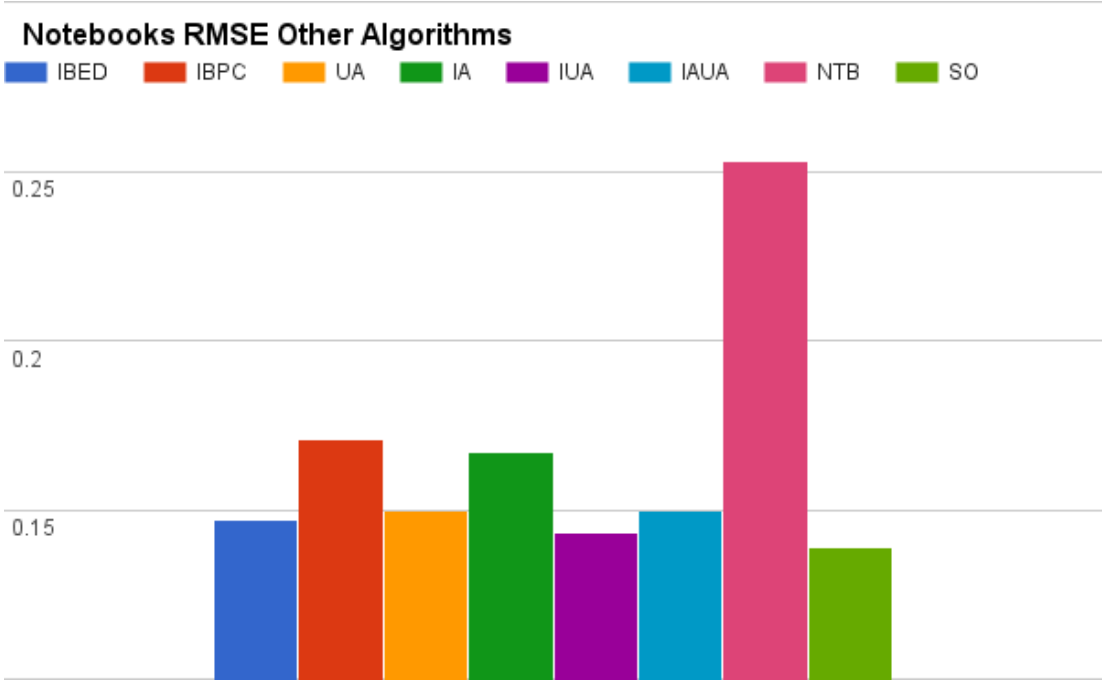


Figure 5.29: Notebooks WRMSE Other Algorithms

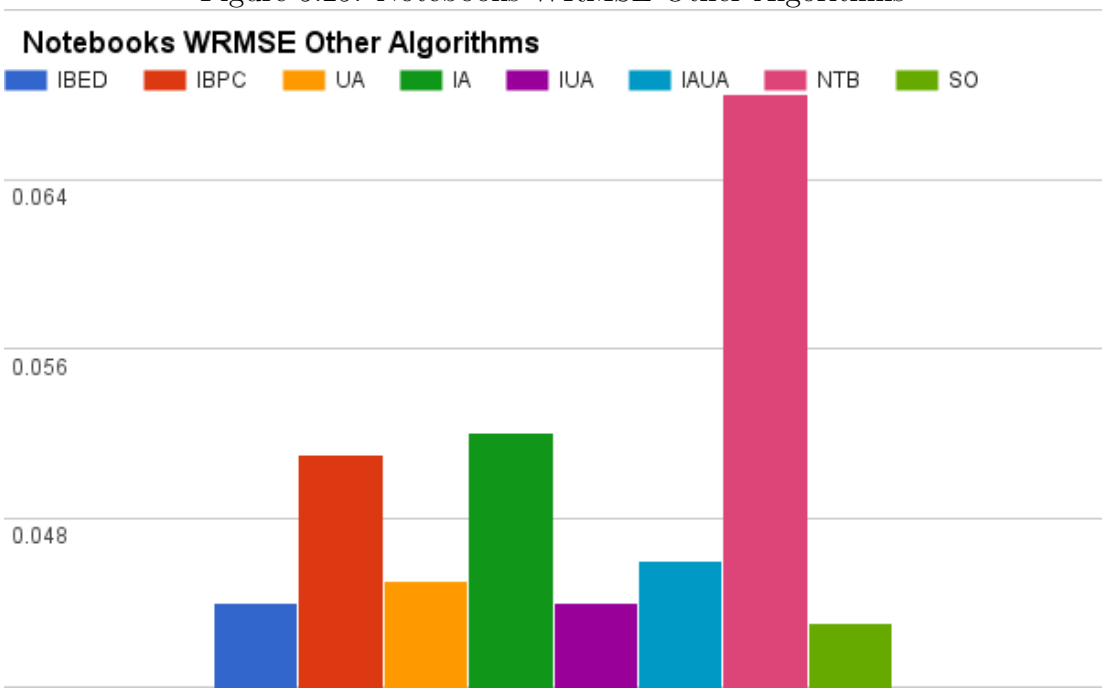
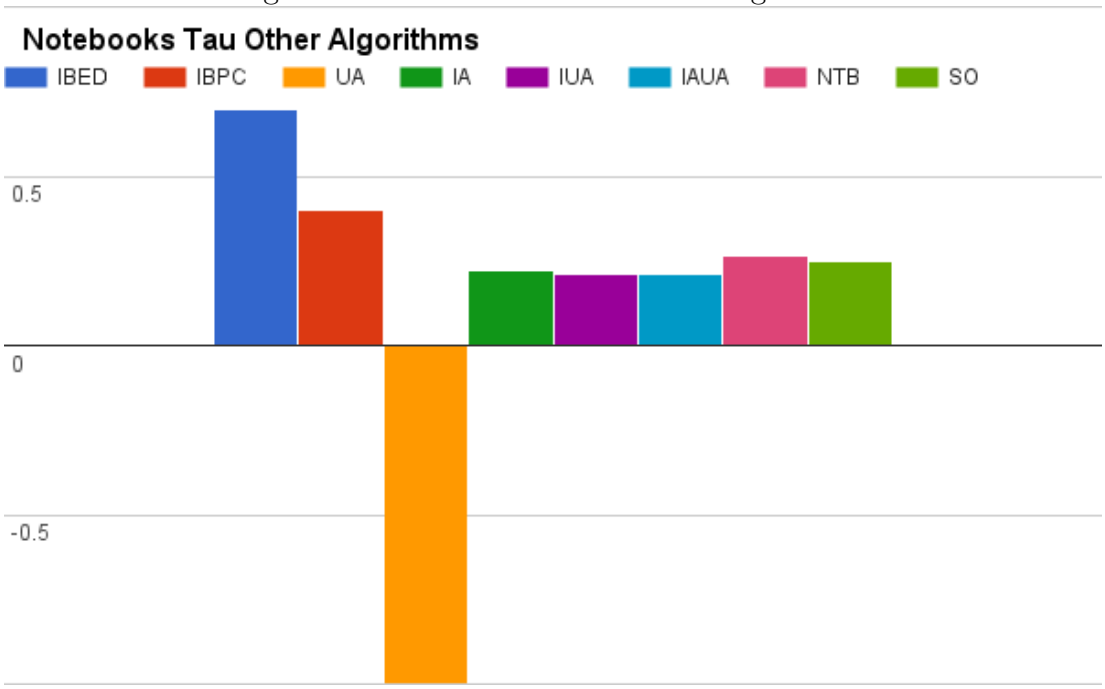


Figure 5.30: Notebooks Tau Other Algorithms



6. Results

6.1 Comparison of models for the same metrics and dataset

To see the results of the models run on already mentioned datasets, please consult chapter 9. In this section we try to show how the selected models performed with fixed dataset and fixed metrics. We see that matrix factorization algorithms performed better than other models. However, these models are not a primary subject of the thesis, therefore we focus on models which use content for prediction. We see that with Movielens dataset, the *UBMLUS* model performed quite well and better than others User Based methods. In the Sushi dataset, the *UBWD* model and *UBSN* model performed quite well again. In the Notebooks dataset we do not see any extraordinary performance of Content Based models.

Figure 6.1: Movielens RMSE best algorithms

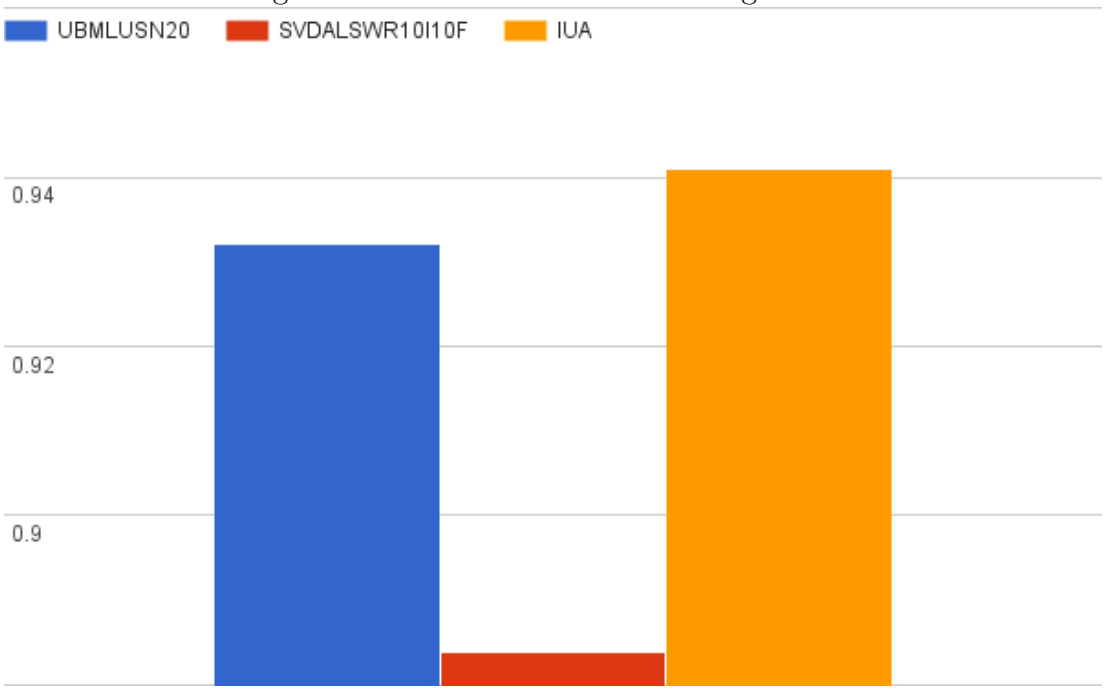


Figure 6.2: Movielens WRMSE best algorithms

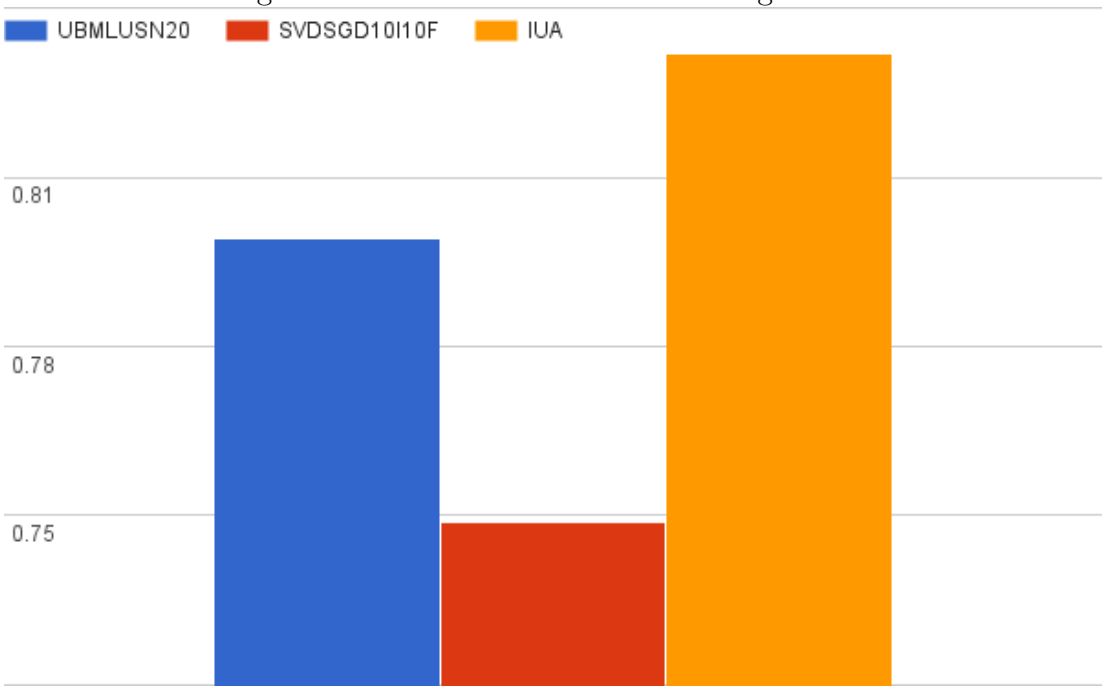


Figure 6.3: Movielens Tau best algorithms

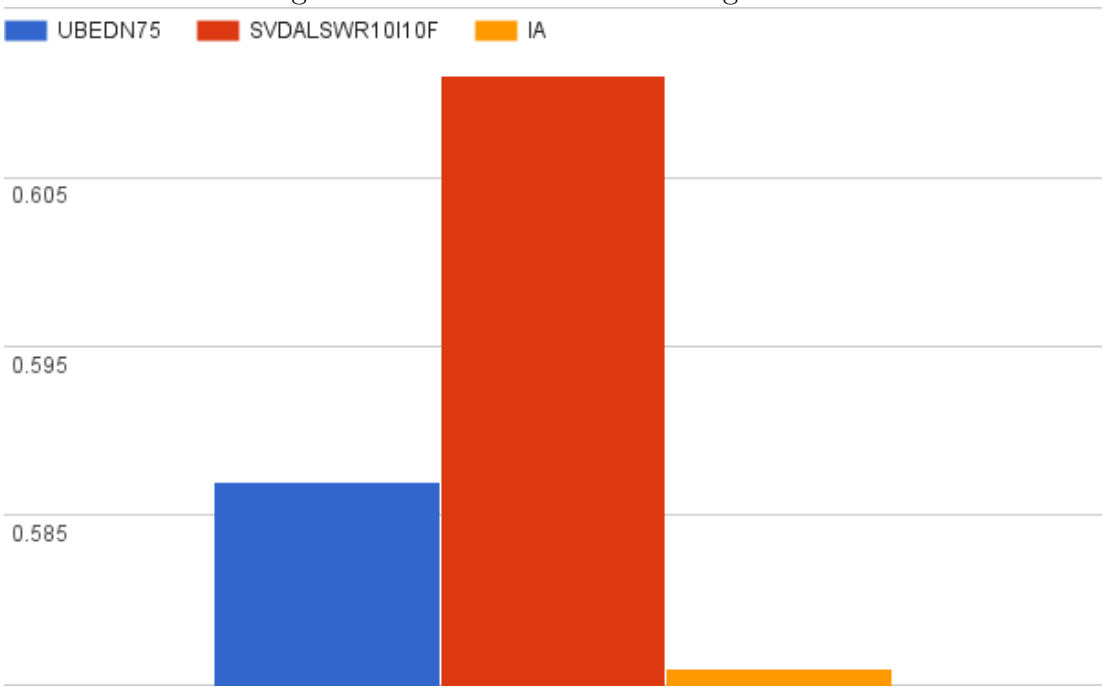


Figure 6.4: Sushi RMSE best algorithms

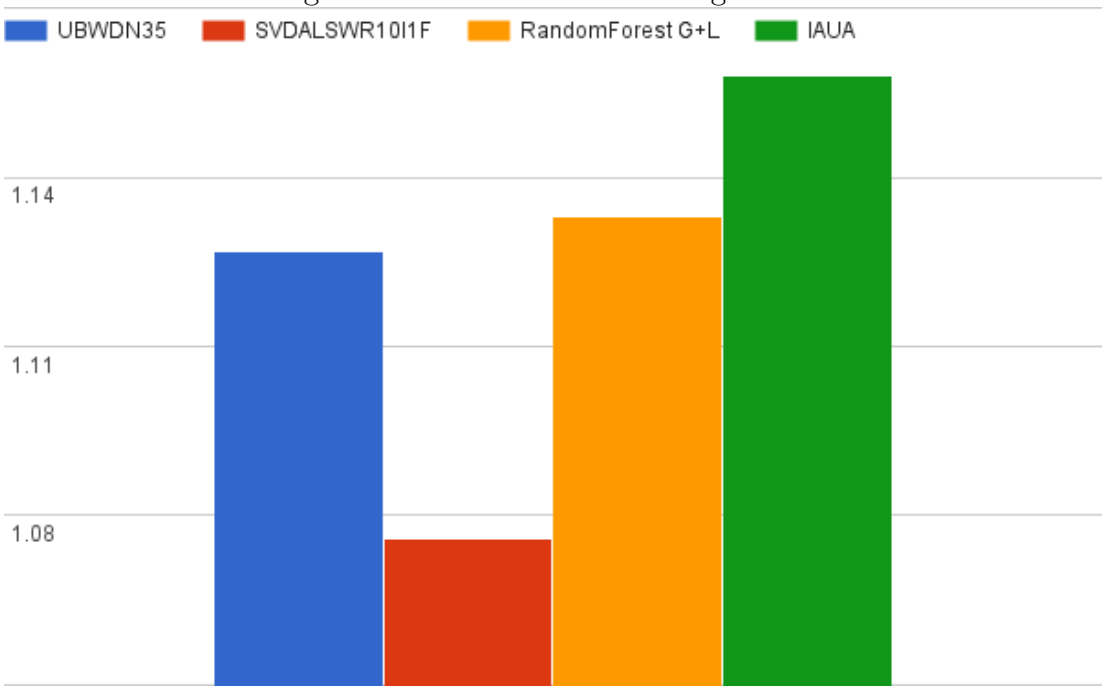


Figure 6.5: Sushi WRMSE best algorithms

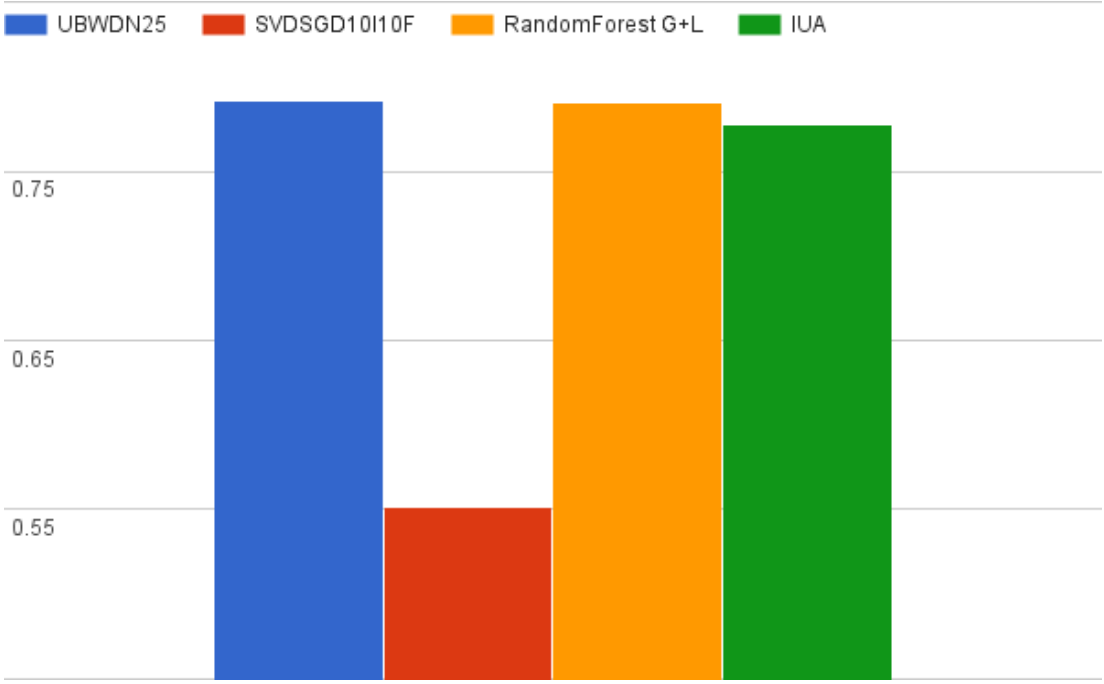


Figure 6.6: Sushi Tau best algorithms

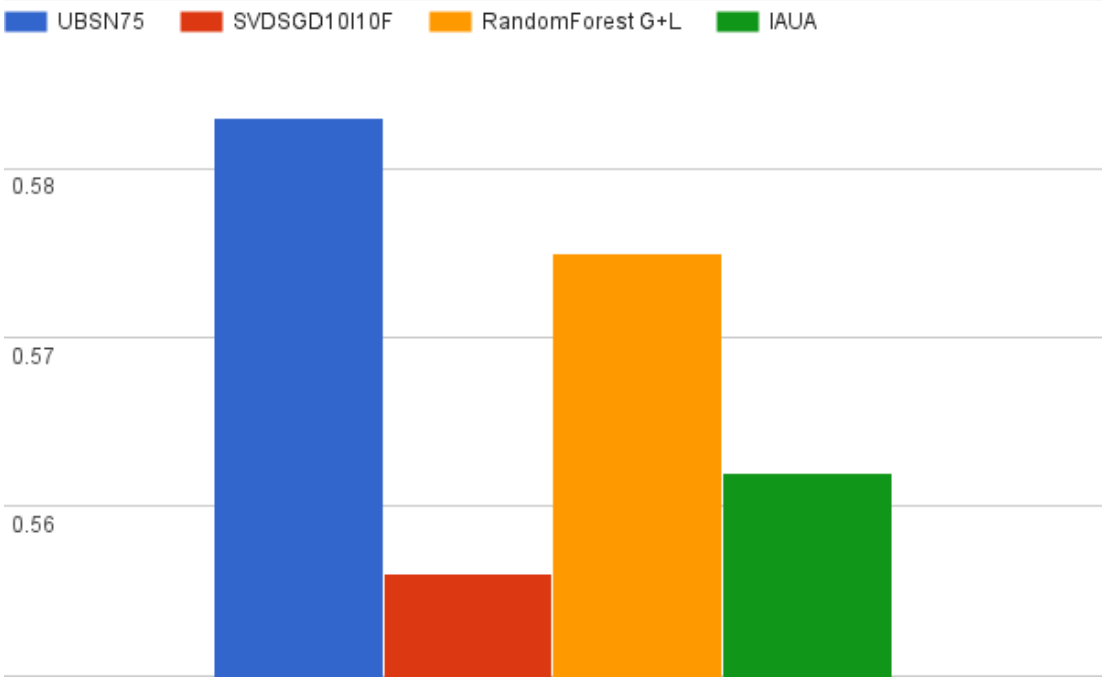


Figure 6.7: Notebooks RMSE best algorithms

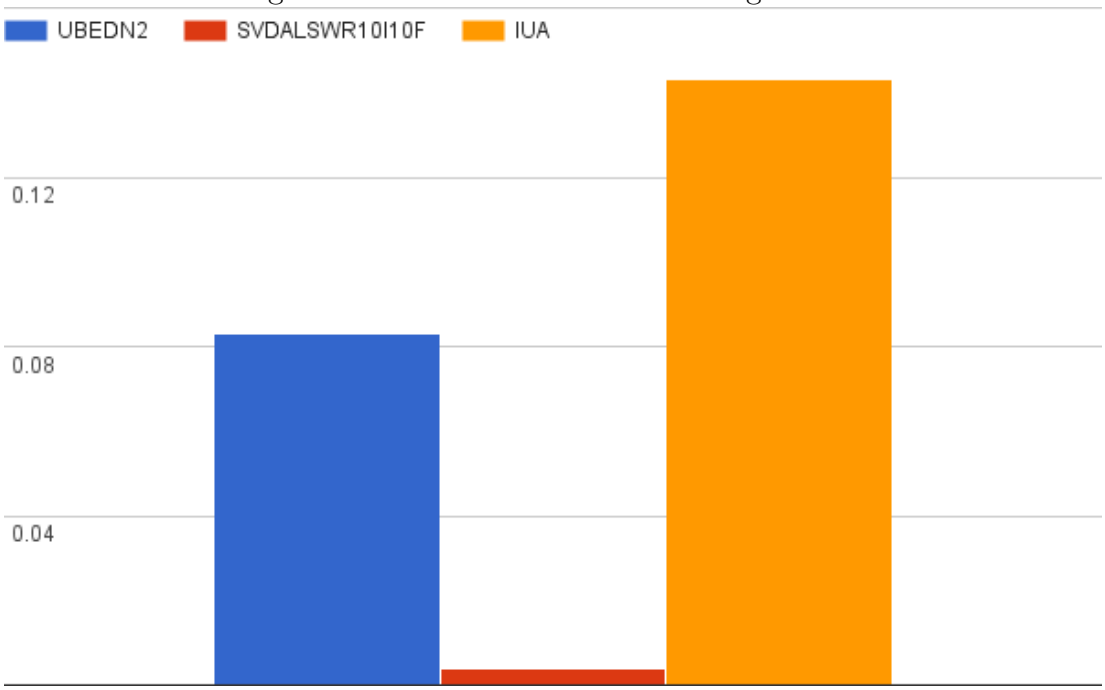


Figure 6.8: Notebooks WRMSE best algorithms

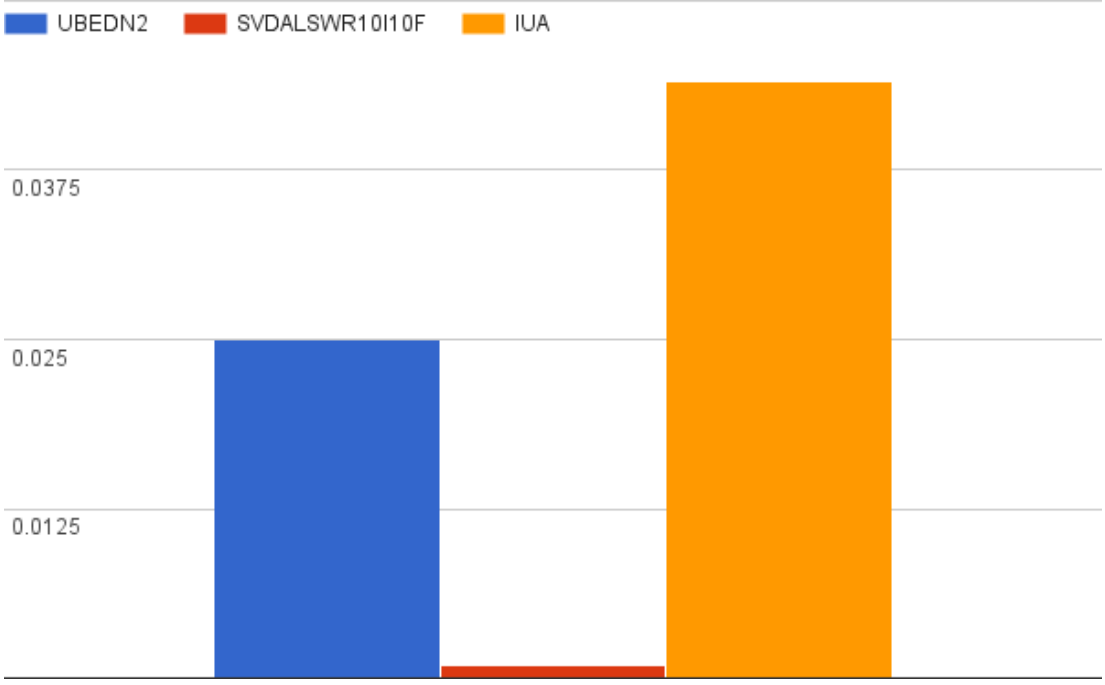
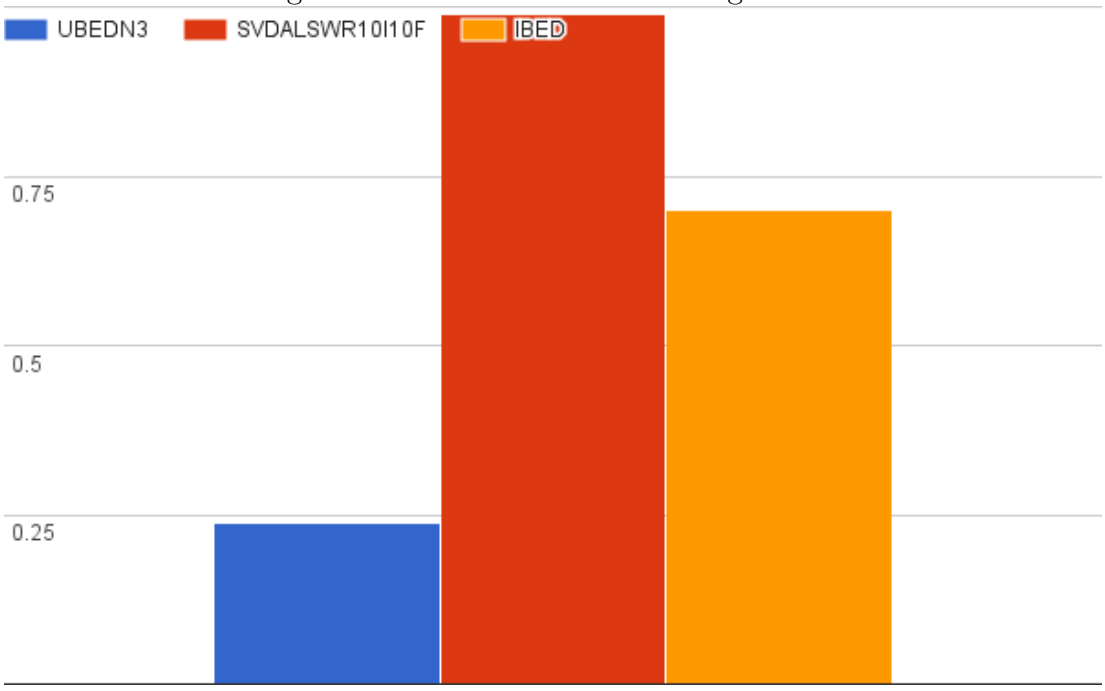


Figure 6.9: Notebooks Tau best algorithms



7. Conclusion

From the charts we can see that content based algorithms sometimes perform better than standard algorithms, sometimes worse. The performance depends on the dataset itself. Even if the content based models performed better, the gain was not very significant. It is impossible to say that content based models or hybrid models are better or worse in general. In the recent article from Fortune.com, Todd Yellin from netflix said that "geography, gender and age are garbage" for prediction, as can be found in [9].

Bibliography

[1] Eckhardt, Alan. *Similarity of users (content-based) preference models for Collaborative filtering in few ratings scenario.* Expert Systems with Applications Volume 39, Issue 14, 15 October 2012, Pages 11511–11516

[2] Amatriain, Xavier, Castells, Pablo, de Vries, Arjen, Posse, Christian. *Workshop on recommendation utility evaluation: beyond RMSE – RUE 2012.* RecSys '12 Proceedings of the sixth ACM conference on Recommender systems Pages 351-352

[3] Yehuda, Koren. *Factorization meets the neighborhood: a multifaceted collaborative filtering model.* KDD '08 Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining Pages 426-434

[4] Zhou, Yunhong, Wilkinson, Dennis, Schreiber, Robert, Pan, Rong. *Large-Scale Parallel Collaborative Filtering for the Netflix Prize.* AAIM '08 Proceedings of the 4th international conference on Algorithmic Aspects in Information and Management Pages 337-348

[5] Cohen W., William, Schapire E., Robert, Singer Yoram *Learning to Order Things* Journal of Artificial Intelligence Research 10 1999, Pages 243-270

[6] Ekstrand D., Michael, Riedl T., John, Konstan A., Joseph *Collaborative Filtering Recommender Systems* Foundations and Trends in Human–Computer Interaction Vol. 4, No. 2 (2010), Pages 81–173

[7] Burges J.C., Christopher, Shaked Tal, Renshaw Erin, Lazier Ari, Deeds Matt, Hamilton Nicole, Hullender Greg *Learning to Rank using Gradient Descent* ICML '05 Proceedings of the 22nd international conference on Machine learning Pages 89-96

[8] Karatzoglou Alexandros, Baltrunas Linas, Shi Yue *Learning to rank for recommender systems* RecSys '13 Proceedings of the 7th ACM conference on Recommender systems Pages 493-494

[9] Morris Z., David *Netflix says Geography, Age, and Gender are "Garbage" for Predicting Taste* Fortune.com. [online]. 27.3.2016, http://fortune.com/2016/03/27/netflix-predicts-taste

[10] Qiu Feng, Cho Junghoo *Automatic identification of user interest for personalized search* WWW '06 Proceedings of the 15th international conference on World Wide Web Pages 727-736

[11] Zhao Yu, Feng Xinping, Li Jiangiang, Liu Bo *Shared collaborative filtering* RecSys '11 Proceedings of the fifth ACM conference on Recommender systems Pages 29-36

8. List of Abbreviations

The following abbreviations can be found in the text of thesis:

- CF - Collaborative Filtering
- CB - Content Based
- RMSE - Root Mean Square Error
- IMDb - The Internet Movie Database
- SVD - Singular Value Decomposition
- ALSWR - Alternating-Least-Squares with Weighted-λ-Regularization
- UBED - User Based Collaborative Filtering with Euclidean Distance
- UBPC - User Based Collaborative Filtering with Pearson Correlation Distance
- IBED - Item Based Collaborative Filtering with Euclidean Distance
- UA - User Average
- IA - Item Average
- IUA - Item User Average
- IAUA - Item And User Average
- SVDSGD - Matrix Factorization using SVD and Stochastic Gradient Descent
- SO - SlopeOne algorithm
- UBMLUS - User Based Movielens User Similarity
- IBMIS - Item Based Movielens Item Similarity
- UBS - User Based with Sushi User Similarity
- UBW - User Based with Sushi User Weights Similarity
- UBWD - User Based with Sushi User Weighted Similarity
- IBSIS - Item Based Sushi Item Similarity
- NTB - Notebooks Recommender

9. Attachments

Sushi RMSE results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBEDN5	449972	28	9	1.271	0.015
UBEDN10	449964	36	9	1.215	0.011
UBEDN15	449954	46	9	1.198	0.012
UBEDN20	449968	32	9	1.191	0.014
UBEDN25	449949	51	10	1.184	0.013
UBEDN30	449968	32	11	1.185	0.013
UBEDN35	449962	38	11	1.183	0.012
UBEDN40	449958	42	11	1.182	0.011
UBEDN45	449964	36	11	1.181	0.013
UBEDN50	449968	32	11	1.181	0.013
UBEDN75	449971	29	13	1.182	0.011
UBEDN100	449967	33	13	1.184	0.012
UBPCN5	411769	38231	8	1.428	0.016
UBPCN10	411347	38653	8	1.509	0.019
UBPCN15	411860	38140	8	1.58	0.023
UBPCN20	411493	38507	9	1.639	0.022
UBPCN25	411227	38773	9	1.672	0.02
UBPCN30	411972	38028	9	1.696	0.022
UBPCN35	411086	38914	9	1.705	0.022
UBPCN40	411725	38275	9	1.714	0.022
UBPCN45	411967	38033	9	1.714	0.02
UBPCN50	411649	38351	9	1.715	0.022
UBPCN75	411425	38575	9	1.717	0.022
UBPCN100	411735	38265	9	1.716	0.024
IBED	449943	57	4	1.243	0.014
IBPC	445633	4367	4	1.547	0.023
UA	450000	0	1	1.253	0.012
IA	450000	0	1	1.198	0.015
IUA	450000	0	1	1.174	0.013
IAUA	450000	0	1	1.158	0.013
gl_J48	450000	0	17	1.188	0.014
gl_RandomTree	450000	0	12	1.159	0.013
gl_Logistic	450000	0	530	1.224	0.015
gl_NaiveBayes	450000	0	11	1.234	0.015
gl_RandomForest	450000	0	558	1.133	0.01
g_J48	450000	0	10	1.215	0.015
g_RandomTree	450000	0	7	1.2	0.012
g_Logistic	450000	0	195	1.228	0.012
g_NaiveBayes	450000	0	6	1.243	0.013
g_RandomForest	450000	0	90	1.198	0.015

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
l_J48	450000	0	10	1.362	0.016
l_RandomTree	450000	0	10	1.373	0.018
l_Logistic	450000	0	333	1.518	0.018
l_NaiveBayes	450000	0	9	1.485	0.016
l_RandomForest	450000	0	480	1.233	0.013
glu_J48	450000	0	56	1.214	0.012
glu_RandomTree	450000	0	21	1.299	0.014
glu_Logistic	450000	0	6823	1.227	0.014
glu_NaiveBayes	450000	0	14	1.238	0.015
glu_RandomForest	450000	0	1178	1.174	0.013
SVDSGD10I4F	450000	0	23	1.448	0.026
SVDSGD10I5F	450000	0	26	1.439	0.026
SVDSGD10I6F	450000	0	30	1.44	0.028
SVDSGD10I7F	450000	0	34	1.435	0.026
SVDSGD10I8F	450000	0	37	1.437	0.027
SVDSGD10I9F	450000	0	41	1.432	0.024
SVDSGD10I10F	450000	0	46	1.439	0.03
SVDSGD10I11F	450000	0	49	1.438	0.025
SVDSGD10I12F	450000	0	54	1.437	0.025
SVDSGD10I13F	450000	0	58	1.436	0.027
SVDALSWR10I1F	450000	0	16	1.076	0.013
SVDALSWR10I2F	450000	0	24	1.079	0.014
SVDALSWR10I3F	450000	0	30	1.078	0.012
SVDALSWR10I4F	450000	0	40	1.08	0.014
SVDALSWR10I5F	450000	0	51	1.082	0.012
SVDALSWR10I6F	450000	0	65	1.085	0.013
SVDALSWR10I7F	450000	0	81	1.086	0.012
SVDALSWR10I8F	450000	0	100	1.088	0.013
SVDALSWR10I9F	450000	0	122	1.088	0.013
SVDALSWR10I10F	450000	0	148	1.087	0.012

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBSN5	440376	9624	89	1.209	0.014
UBSN10	440181	9819	90	1.167	0.015
UBSN15	440538	9462	93	1.153	0.014
UBSN20	440187	9813	95	1.142	0.012
UBSN25	440304	9696	99	1.138	0.014
UBSN30	440349	9651	99	1.136	0.012
UBSN35	440139	9861	99	1.132	0.012
UBSN40	440370	9630	100	1.131	0.012
UBSN45	440307	9693	103	1.13	0.012
UBSN50	440379	9621	104	1.129	0.013
UBSN75	440121	9879	112	1.133	0.012
UBSN100	440553	9447	120	1.14	0.012
UBWN5	450000	0	195	1.255	0.015
UBWN10	450000	0	197	1.208	0.014
UBWN15	450000	0	200	1.193	0.012
UBWN20	450000	0	205	1.186	0.013
UBWN25	450000	0	209	1.179	0.014
UBWN30	450000	0	213	1.179	0.013
UBWN35	450000	0	218	1.176	0.012
UBWN40	450000	0	222	1.177	0.014
UBWN45	450000	0	227	1.177	0.011
UBWN50	450000	0	231	1.177	0.011
UBWN75	450000	0	249	1.175	0.011
UBWN100	450000	0	268	1.178	0.011
IBSIS	450000	0	1	1.236	0.012
UBWDN5	440298	9702	661	1.201	0.015
UBWDN10	440430	9570	666	1.16	0.012
UBWDN15	440271	9729	679	1.145	0.013
UBWDN20	440178	9822	684	1.138	0.012
UBWDN25	440325	9675	693	1.133	0.011
UBWDN30	440403	9597	695	1.131	0.012
UBWDN35	440547	9453	694	1.127	0.012
UBWDN40	440481	9519	701	1.128	0.013
UBWDN45	440235	9765	705	1.128	0.012
UBWDN50	440094	9906	707	1.128	0.011
UBWDN75	440316	9684	713	1.129	0.011
UBWDN100	440328	9672	716	1.133	0.013
IBSIS	450000	0	1	1.237	0.014
SO	450000	0	2	1.192	0.012

Sushi WRMSE results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBEDN5	449970	30	9	0.884	0.018
UBEDN10	449976	24	10	0.865	0.018
UBEDN15	449960	40	9	0.863	0.018
UBEDN20	449973	27	10	0.86	0.015
UBEDN25	449964	36	10	0.855	0.015
UBEDN30	449968	32	10	0.858	0.017
UBEDN35	449956	44	11	0.858	0.018
UBEDN40	449957	43	11	0.851	0.017
UBEDN45	449973	27	12	0.853	0.016
UBEDN50	449971	29	12	0.854	0.017
UBEDN75	449966	34	13	0.856	0.016
UBEDN100	449959	41	14	0.857	0.018
UBPCN5	411709	38291	8	0.917	0.019
UBPCN10	411798	38202	8	0.951	0.019
UBPCN15	411758	38242	8	0.977	0.02
UBPCN20	411345	38655	9	0.992	0.019
UBPCN25	412123	37877	9	1.004	0.021
UBPCN30	411586	38414	9	1.006	0.02
UBPCN35	411340	38660	9	1.013	0.021
UBPCN40	411469	38531	9	1.014	0.02
UBPCN45	411565	38435	9	1.012	0.019
UBPCN50	411631	38369	9	1.015	0.021
UBPCN75	411561	38439	9	1.014	0.019
UBPCN100	411969	38031	9	1.012	0.02
IBED	449939	61	4	0.838	0.015
IBPC	445519	4481	4	0.914	0.021
UA	450000	0	1	0.84	0.016
IA	450000	0	1	0.854	0.015
IUA	450000	0	1	0.778	0.016
IAUA	450000	0	1	0.818	0.016
gl_J48	450000	0	17	0.832	0.018
gl_RandomTree	450000	0	12	0.805	0.017
gl_Logistic	450000	0	524	0.829	0.019
gl_NaiveBayes	450000	0	10	0.878	0.016
gl_RandomForest	450000	0	556	0.791	0.017
g_J48	450000	0	10	0.87	0.02
g_RandomTree	450000	0	6	0.859	0.016
g_Logistic	450000	0	196	0.881	0.017
g_NaiveBayes	450000	0	6	0.889	0.014

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
g_RandomForest	450000	0	91	0.854	0.015
l_J48	450000	0	10	0.885	0.019
l_RandomTree	450000	0	9	0.875	0.018
l_Logistic	450000	0	329	0.934	0.019
l_NaiveBayes	450000	0	9	0.993	0.022
l_RandomForest	450000	0	473	0.806	0.019
glu_J48	450000	0	56	0.846	0.015
glu_RandomTree	450000	0	20	0.855	0.019
glu_Logistic	450000	0	7335	0.826	0.019
glu_NaiveBayes	450000	0	14	0.879	0.02
glu_RandomForest	450000	0	1173	0.801	0.017
SVDSGD10I4F	450000	0	23	0.555	0.009
SVDSGD10I5F	450000	0	25	0.553	0.009
SVDSGD10I6F	450000	0	29	0.553	0.008
SVDSGD10I7F	450000	0	33	0.552	0.007
SVDSGD10I8F	450000	0	37	0.553	0.009
SVDSGD10I9F	450000	0	41	0.552	0.009
SVDSGD10I10F	450000	0	45	0.552	0.008
SVDSGD10I11F	450000	0	50	0.554	0.009
SVDSGD10I12F	450000	0	53	0.552	0.009
SVDSGD10I13F	450000	0	58	0.552	0.008
SVDALSWR10I1F	450000	0	15	0.821	0.014
SVDALSWR10I2F	450000	0	23	0.815	0.016
SVDALSWR10I3F	450000	0	30	0.815	0.017
SVDALSWR10I4F	450000	0	39	0.816	0.014
SVDALSWR10I5F	450000	0	50	0.814	0.015
SVDALSWR10I6F	450000	0	64	0.814	0.016
SVDALSWR10I7F	450000	0	80	0.817	0.015
SVDALSWR10I8F	450000	0	98	0.817	0.013
SVDALSWR10I9F	450000	0	120	0.814	0.016
SVDALSWR10I10F	450000	0	145	0.818	0.013
UBSN5	440535	9465	93	0.817	0.02

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBSN10	440013	9987	93	0.805	0.019
UBSN15	440310	9690	96	0.799	0.016
UBSN20	440253	9747	99	0.797	0.016
UBSN25	440265	9735	100	0.796	0.017
UBSN30	440106	9894	102	0.8	0.016
UBSN35	440307	9693	105	0.804	0.016
UBSN40	440448	9552	107	0.804	0.017
UBSN45	440370	9630	109	0.806	0.018
UBSN50	439899	10101	111	0.809	0.017
UBSN75	440097	9903	120	0.809	0.017
UBSN100	440493	9507	127	0.821	0.016
UBWN5	450000	0	192	0.857	0.017
UBWN10	450000	0	196	0.837	0.018
UBWN15	450000	0	200	0.835	0.017
UBWN20	450000	0	206	0.831	0.018
UBWN25	450000	0	217	0.832	0.015
UBWN30	450000	0	215	0.831	0.016
UBWN35	450000	0	218	0.83	0.018
UBWN40	450000	0	222	0.833	0.017
UBWN45	450000	0	226	0.835	0.019
UBWN50	450000	0	231	0.836	0.016
UBWN75	450000	0	249	0.836	0.016
UBWN100	450000	0	268	0.842	0.015
UBWDN5	440190	9810	672	0.811	0.017
UBWDN10	440433	9567	676	0.798	0.018
UBWDN15	440193	9807	684	0.798	0.017
UBWDN20	440271	9729	692	0.794	0.016
UBWDN25	440349	9651	696	0.792	0.016
UBWDN30	440349	9651	700	0.798	0.015
UBWDN35	440001	9999	706	0.803	0.018
UBWDN40	440148	9852	707	0.798	0.015
UBWDN45	440412	9588	709	0.799	0.016
UBWDN50	440286	9714	713	0.806	0.018
UBWDN75	440352	9648	718	0.809	0.015
UBWDN100	440322	9678	721	0.815	0.017
IBSIS	450000	0	1	0.833	0.016
SO	450000	0	2	0.788	0.016

Sushi Tau results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBEDN5	449964	36	9	0.398	0.017
UBEDN10	449966	34	9	0.484	0.015
UBEDN15	449968	32	9	0.517	0.014
UBEDN20	449964	36	10	0.531	0.015
UBEDN25	449959	41	10	0.543	0.013
UBEDN30	449965	35	10	0.553	0.012
UBEDN35	449958	42	11	0.556	0.013
UBEDN40	449960	40	11	0.56	0.012
UBEDN45	449971	29	11	0.561	0.014
UBEDN50	449959	41	12	0.565	0.012
UBEDN75	449950	50	13	0.566	0.014
UBEDN100	449971	29	14	0.567	0.014
UBPCN5	411927	38073	8	0.379	0.018
UBPCN10	411540	38460	8	0.4	0.015
UBPCN15	411407	38593	8	0.379	0.02
UBPCN20	412266	37734	9	0.357	0.019
UBPCN25	411430	38570	9	0.343	0.017
UBPCN30	412130	37870	9	0.331	0.02
UBPCN35	411541	38459	9	0.324	0.02
UBPCN40	411934	38066	9	0.324	0.019
UBPCN45	412082	37918	9	0.319	0.019
UBPCN50	411430	38570	9	0.322	0.02
UBPCN75	411445	38555	9	0.326	0.02
UBPCN100	411454	38546	9	0.321	0.017
IBED	449950	50	5	0.381	0.018
IBPC	445599	4401	5	0.28	0.018
UA	450000	0	1	-1	0
IA	450000	0	1	0.561	0.013
IUA	450000	0	1	0.536	0.015
IAUA	450000	0	1	0.562	0.014
gl_J48	450000	0	17	0.414	0.062
gl_RandomTree	450000	0	12	0.542	0.014
gl_Logistic	450000	0	530	0.478	0.016
gl_NaiveBayes	450000	0	11	0.478	0.014
gl_RandomForest	450000	0	559	0.575	0.011
g_J48	450000	0	10	0.415	0.086
g_RandomTree	450000	0	7	0.56	0.015
g_Logistic	450000	0	196	0.504	0.013
g_NaiveBayes	450000	0	6	0.499	0.013

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
g_RandomForest	450000	0	92	0.562	0.014
l_J48	450000	0	10	-0.564	0.014
l_RandomTree	450000	0	10	-0.148	0.018
l_Logistic	450000	0	335	0.238	0.018
l_NaiveBayes	450000	0	9	0.145	0.019
l_RandomForest	450000	0	477	0.368	0.015
glu_J48	450000	0	56	0.32	0.063
glu_RandomTree	450000	0	21	0.287	0.024
glu_Logistic	450000	0	7376	0.477	0.015
glu_NaiveBayes	450000	0	14	0.479	0.016
glu_RandomForest	450000	0	1165	0.502	0.014
SVDSGD10I4F	450000	0	22	0.556	0.015
SVDSGD10I5F	450000	0	25	0.554	0.013
SVDSGD10I6F	450000	0	29	0.554	0.014
SVDSGD10I7F	450000	0	33	0.553	0.013
SVDSGD10I8F	450000	0	37	0.554	0.011
SVDSGD10I9F	450000	0	41	0.552	0.014
SVDSGD10I10F	450000	0	45	0.555	0.014
SVDSGD10I11F	450000	0	49	0.556	0.014
SVDSGD10I12F	450000	0	53	0.554	0.014
SVDSGD10I13F	450000	0	58	0.556	0.015
SVDALSWR10I1F	450000	0	16	0.541	0.013
SVDALSWR10I2F	450000	0	23	0.538	0.013
SVDALSWR10I3F	450000	0	31	0.536	0.014
SVDALSWR10I4F	450000	0	40	0.533	0.014
SVDALSWR10I5F	450000	0	52	0.531	0.014
SVDALSWR10I6F	450000	0	66	0.529	0.015
SVDALSWR10I7F	450000	0	83	0.526	0.013
SVDALSWR10I8F	450000	0	102	0.525	0.014
SVDALSWR10I9F	450000	0	125	0.52	0.013
SVDALSWR10I10F	450000	0	151	0.521	0.014

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBSN5	440430	9570	90	0.504	0.013
UBSN10	440190	9810	90	0.546	0.016
UBSN15	440127	9873	93	0.561	0.012
UBSN20	440262	9738	95	0.565	0.014
UBSN25	440388	9612	97	0.568	0.014
UBSN30	440346	9654	99	0.571	0.014
UBSN35	440445	9555	100	0.575	0.015
UBSN40	440385	9615	101	0.577	0.014
UBSN45	440367	9633	103	0.577	0.014
UBSN50	440409	9591	105	0.579	0.014
UBSN75	440526	9474	114	0.583	0.012
UBSN100	440319	9681	124	0.577	0.013
UBWN5	450000	0	192	0.481	0.014
UBWN10	450000	0	196	0.52	0.014
UBWN15	450000	0	200	0.537	0.014
UBWN20	450000	0	205	0.546	0.013
UBWN25	450000	0	209	0.547	0.013
UBWN30	450000	0	213	0.55	0.015
UBWN35	450000	0	217	0.556	0.015
UBWN40	450000	0	222	0.557	0.015
UBWN45	450000	0	227	0.559	0.013
UBWN50	450000	0	230	0.561	0.013
UBWN75	450000	0	250	0.562	0.014
UBWN100	450000	0	269	0.56	0.013
UBWDN5	440169	9831	706	0.514	0.014
UBWDN10	440400	9600	704	0.55	0.013
UBWDN15	440055	9945	707	0.561	0.014
UBWDN20	440145	9855	707	0.568	0.015
UBWDN25	440298	9702	707	0.572	0.012
UBWDN30	440268	9732	709	0.573	0.014
UBWDN35	440355	9645	710	0.577	0.015
UBWDN40	440319	9681	709	0.578	0.012
UBWDN45	440424	9576	715	0.577	0.013
UBWDN50	440172	9828	719	0.579	0.014
UBWDN75	440319	9681	721	0.58	0.013
UBWDN100	440394	9606	718	0.581	0.013
IBSIS	450000	0	1	0.422	0.014
SO	450000	0	3	0.523	0.014

Movielens RMSE results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
mlcb	8630714	417110	1677	1.012	0.005
UBEDN5	9047439	18977	1382	1.003	0.007
UBEDN10	9065284	18853	1451	0.961	0.007
UBEDN15	9086573	18974	1511	0.949	0.006
UBEDN20	9079428	19136	1539	0.946	0.006
UBEDN25	9081594	19033	1529	0.944	0.006
UBEDN30	9101176	19114	1583	0.943	0.007
UBEDN35	9081315	19186	1615	0.945	0.006
UBEDN40	9053606	19321	1642	0.945	0.006
UBEDN45	9088515	19247	1709	0.947	0.005
UBEDN50	9045259	19298	1723	0.948	0.006
UBEDN75	9056400	18951	1900	0.953	0.006
UBEDN100	9059425	18903	2051	0.959	0.006
UBPCN5	9050292	20777	1354	1.051	0.006
UBPCN10	9087353	20824	1405	1.01	0.006
UBPCN15	9064703	20808	1446	0.996	0.006
UBPCN20	9079826	20631	1496	0.992	0.006
UBPCN25	9080445	20860	1538	0.991	0.007
UBPCN30	9033858	20943	1569	0.99	0.007
UBPCN35	9060484	20544	1624	0.991	0.007
UBPCN40	9100473	20871	1681	0.992	0.008
UBPCN45	9057973	20491	1703	0.993	0.008
UBPCN50	9027630	21118	1735	0.995	0.007
UBPCN75	9038542	20861	1916	0.999	0.007
UBPCN100	9043718	20803	2051	1.004	0.008
IBED	9104177	8054	1330	1.016	0.005
UA	9069550	0	35	1.037	0.005
IUA	9082872	7874	20	0.941	0.006
IAUA	9111116	7866	20	0.959	0.005
IA	9037734	7966	18	0.986	0.007
SO	9080465	8029	634	0.908	0.005

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBMLUSN5	9073926	19140	22030	0.968	0.006
UBMLUSN10	9067449	18941	22729	0.939	0.006
UBMLUSN15	9075436	18843	23050	0.934	0.005
UBMLUSN20	9080377	18462	23313	0.932	0.005
UBMLUSN25	9079063	18586	23370	0.932	0.005
UBMLUSN30	9086512	19051	23394	0.933	0.006
UBMLUSN35	9064057	19019	23246	0.935	0.007
UBMLUSN40	9063882	18663	23579	0.937	0.006
UBMLUSN45	9069083	18739	23551	0.937	0.005
UBMLUSN50	9055605	18709	23586	0.939	0.005
UBMLUSN75	9085193	18571	24055	0.946	0.005
UBMLUSN100	9035651	19003	23814	0.95	0.006
SVDSGD10I4F	9081593	0	4586	0.966	0.011
SVDSGD10I5F	9088855	0	6174	0.963	0.01
SVDSGD10I6F	9064033	0	7935	0.964	0.01
SVDSGD10I7F	9089719	0	9742	0.964	0.011
SVDSGD10I8F	9042663	0	11566	0.963	0.01
SVDSGD10I9F	9076592	0	13314	0.962	0.01
SVDSGD10I10F	9089861	0	15408	0.959	0.012
SVDSGD10I11F	9080303	0	17256	0.963	0.012
SVDSGD10I12F	9113549	0	18962	0.959	0.01
SVDSGD10I13F	9113079	0	20658	0.961	0.01
SVDALSWR10I1F	9087428	0	127	0.903	0.006
SVDALSWR10I2F	9078108	0	153	0.9	0.006
SVDALSWR10I3F	9106284	0	164	0.898	0.005
SVDALSWR10I4F	9075475	0	196	0.895	0.005
SVDALSWR10I5F	9111511	0	222	0.893	0.005
SVDALSWR10I6F	9070663	0	255	0.891	0.005
SVDALSWR10I7F	9091085	0	295	0.89	0.005
SVDALSWR10I8F	9116165	0	339	0.887	0.006
SVDALSWR10I9F	9065473	0	392	0.885	0.006
SVDALSWR10I10F	9072127	0	454	0.884	0.005
IBPC	9054117	29213	1413	1.042	0.011
IBMIS	8602714	481924	1845	1.012	0.005

Movielens WRMSE results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
mlcb	8656616	418102	1672	0.88	0.004
UBEDN5	9047994	19289	1369	0.854	0.005
UBEDN10	9059474	18798	1418	0.814	0.005
UBEDN15	9043277	19219	1464	0.803	0.005
UBEDN20	9042310	19072	1498	0.799	0.005
UBEDN25	9036133	18875	1539	0.798	0.004
UBEDN30	9077989	18992	1599	0.799	0.004
UBEDN35	9047542	19223	1633	0.801	0.004
UBEDN40	9054961	19015	1677	0.803	0.004
UBEDN45	9062644	19174	1715	0.804	0.004
UBEDN50	9083441	19455	1766	0.805	0.005
UBEDN75	9048050	18877	1936	0.814	0.004
UBEDN100	9069908	18798	2107	0.822	0.004
UBPCN5	9041503	20881	1385	0.924	0.006
UBPCN10	9046270	20847	1434	0.885	0.006
UBPCN15	9071988	20793	1488	0.873	0.005
UBPCN20	9063643	21074	1533	0.87	0.006
UBPCN25	9056762	21259	1576	0.868	0.006
UBPCN30	9049978	20783	1614	0.868	0.006
UBPCN35	9075635	21094	1675	0.868	0.006
UBPCN40	9086113	20798	1726	0.869	0.006
UBPCN45	9040773	20929	1754	0.872	0.006
UBPCN50	9048392	21091	1787	0.872	0.006
UBPCN75	9091626	20857	1988	0.877	0.006
UBPCN100	9064437	20727	2127	0.883	0.006
IBED	9094991	7954	1384	0.881	0.004
UA	9109969	0	34	0.905	0.005
IUA	9052607	7900	18	0.832	0.005
IAUA	9057618	8038	18	0.833	0.004
IA	9108112	7643	17	0.859	0.006
SO	9070983	8012	625	0.798	0.004

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBMLUSN5	9060240	18977	255922	0.84	0.004
UBMLUSN10	9068395	19030	23065	0.811	0.004
UBMLUSN15	9048688	19194	22747	0.801	0.004
UBMLUSN20	9056269	19072	22738	0.799	0.005
UBMLUSN25	9035314	19022	22829	0.799	0.005
UBMLUSN30	9065320	18922	23178	0.8	0.005
UBMLUSN35	9070784	18981	23189	0.8	0.005
UBMLUSN40	9045341	18638	23062	0.802	0.004
UBMLUSN45	9079778	18673	23583	0.802	0.004
UBMLUSN50	9061416	18803	23476	0.804	0.005
UBMLUSN75	9041214	18685	23447	0.811	0.004
UBMLUSN100	9067631	18787	23963	0.816	0.005
SVDSGD10I4F	9070396	0	4593	0.753	0.004
SVDSGD10I5F	9060019	0	6224	0.753	0.005
SVDSGD10I6F	9097670	0	7995	0.752	0.004
SVDSGD10I7F	9078609	0	9843	0.751	0.005
SVDSGD10I8F	9110008	0	11663	0.751	0.005
SVDSGD10I9F	9061350	0	13520	0.751	0.005
SVDSGD10I10F	9080604	0	15359	0.751	0.005
SVDSGD10I11F	9064424	0	17168	0.75	0.005
SVDSGD10I12F	9104913	0	18911	0.749	0.005
SVDSGD10I13F	9101891	0	20725	0.749	0.004
SVDALSWR10I1F	9076893	0	133	0.797	0.005
SVDALSWR10I2F	9081813	0	155	0.794	0.004
SVDALSWR10I3F	9082048	0	176	0.792	0.004
SVDALSWR10I4F	9088981	0	201	0.79	0.004
SVDALSWR10I5F	9063440	0	229	0.788	0.004
SVDALSWR10I6F	9065485	0	265	0.787	0.004
SVDALSWR10I7F	9082329	0	305	0.785	0.004
SVDALSWR10I8F	9059288	0	349	0.784	0.004
SVDALSWR10I9F	9089488	0	404	0.782	0.004
SVDALSWR10I10F	9053069	0	465	0.781	0.005
IBPC	9046172	28811	1371	0.913	0.011
IBMIS	8569570	478238	1820	0.887	0.004

Movielens Tau results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
mlcb	8667163	419363	1662	0.413	0.004
UBEDN5	9087658	19102	1398	0.493	0.004
UBEDN10	9063572	18946	1431	0.544	0.004
UBEDN15	9060124	19287	1484	0.562	0.004
UBEDN20	9085152	19141	1529	0.572	0.004
UBEDN25	9028706	18964	1548	0.577	0.004
UBEDN30	9060229	18900	1604	0.58	0.004
UBEDN35	9056404	19294	1646	0.583	0.004
UBEDN40	9056565	18775	1693	0.584	0.004
UBEDN45	9068605	19071	1734	0.584	0.004
UBEDN50	9069333	19340	1770	0.585	0.004
UBEDN75	9074419	18882	1950	0.587	0.004
UBEDN100	9054732	18614	2097	0.586	0.004
UBPCN5	9020068	20979	1342	0.494	0.005
UBPCN10	9076404	20736	1410	0.54	0.004
UBPCN15	9071535	21241	1455	0.556	0.004
UBPCN20	9052793	20729	1499	0.563	0.004
UBPCN25	9066480	20894	1554	0.567	0.004
UBPCN30	9056406	20885	1592	0.569	0.004
UBPCN35	9085777	21152	1645	0.57	0.005
UBPCN40	9098843	20974	1692	0.569	0.004
UBPCN45	9037512	21197	1711	0.569	0.004
UBPCN50	9043491	21110	1745	0.569	0.004
UBPCN75	9064837	21163	1930	0.568	0.004
UBPCN100	9080939	20696	2086	0.565	0.004
IBED	9085548	8269	1366	0.539	0.006
UA	9110500	0	35	-1	0
IUA	9048866	7986	20	0.558	0.004
IAUA	9065344	7836	20	0.575	0.004
IA	9047449	7904	19	0.576	0.004
SO	9072569	7878	624	0.585	0.004

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UBMLUSN5	9028771	18537	21766	0.523	0.005
UBMLUSN10	9102407	19138	23103	0.557	0.004
UBMLUSN15	9071548	18883	22879	0.569	0.004
UBMLUSN20	9065643	19175	22991	0.575	0.004
UBMLUSN25	9110041	18716	23357	0.579	0.004
UBMLUSN30	9048954	18837	23106	0.581	0.004
UBMLUSN35	9083900	18540	23441	0.583	0.004
UBMLUSN40	9048970	18721	23188	0.583	0.004
UBMLUSN45	9067688	18821	23658	0.584	0.003
UBMLUSN50	9069511	18579	23749	0.584	0.004
UBMLUSN75	9091239	19041	24154	0.584	0.004
UBMLUSN100	9072206	18587	24575	0.583	0.004
SVDSGD10I4F	9095317	0	4628	0.464	0.01
SVDSGD10I5F	9087655	0	6203	0.465	0.01
SVDSGD10I6F	9081751	0	7996	0.465	0.011
SVDSGD10I7F	9088067	0	9719	0.466	0.011
SVDSGD10I8F	9099010	0	11509	0.467	0.01
SVDSGD10I9F	9085773	0	13329	0.467	0.01
SVDSGD10I10F	9077925	0	15113	0.466	0.011
SVDSGD10I11F	9074865	0	16949	0.469	0.009
SVDSGD10I12F	9099736	0	18809	0.468	0.008
SVDSGD10I13F	9065688	0	20562	0.468	0.011
SVDALSWR10I1F	9119966	0	130	0.584	0.004
SVDALSWR10I2F	9074526	0	155	0.587	0.004
SVDALSWR10I3F	9077123	0	182	0.591	0.004
SVDALSWR10I4F	9092102	0	202	0.594	0.003
SVDALSWR10I5F	9097663	0	228	0.597	0.003
SVDALSWR10I6F	9091571	0	262	0.6	0.004
SVDALSWR10I7F	9123735	0	302	0.603	0.004
SVDALSWR10I8F	9069360	0	348	0.606	0.004
SVDALSWR10I9F	9117081	0	401	0.608	0.004
SVDALSWR10I10F	9086843	0	465	0.611	0.003
IBPC	9057429	29516	1403	0.425	0.007
IBMIS	8624863	484973	1844	0.404	0.004

Notebooks RMSE results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
NTB	192000	0	1	0.253	0.009
UBShdpmrN2	192000	0	12	0.104	0.005
UBShdpmrN3	192000	0	13	0.102	0.005
UBShdpmrN4	192000	0	14	0.105	0.005
UBShdpmrN5	192000	0	13	0.108	0.005
UBShdpmrN6	192000	0	13	0.111	0.005
UBShdpmrN7	192000	0	13	0.113	0.005
UBShdpmrN8	192000	0	15	0.116	0.005
UBShdpmrN9	192000	0	14	0.119	0.004
UBShdpmrN10	192000	0	16	0.123	0.004
UBShdpmrN15	192000	0	17	0.14	0.004
UBShdpmrN20	192000	0	18	0.157	0.005
UBShdpmrN30	192000	0	23	0.164	0.006
UBEDN2	192000	0	3	0.083	0.003
UBEDN3	192000	0	3	0.086	0.003
UBEDN4	192000	0	3	0.089	0.003
UBEDN5	192000	0	3	0.094	0.003
UBEDN6	192000	0	3	0.096	0.003
UBEDN7	192000	0	3	0.102	0.003
UBEDN8	192000	0	3	0.105	0.003
UBEDN9	192000	0	3	0.109	0.003
UBEDN10	192000	0	3	0.113	0.003
UBEDN15	192000	0	4	0.135	0.004
UBEDN20	192000	0	4	0.155	0.005
UBEDN30	192000	0	5	0.162	0.006
UBPCN2	192000	0	3	0.088	0.006
UBPCN3	192000	0	3	0.091	0.005
UBPCN4	192000	0	3	0.093	0.005
UBPCN5	192000	0	3	0.097	0.004
UBPCN6	192000	0	3	0.099	0.005
UBPCN7	192000	0	3	0.102	0.005
UBPCN8	192000	0	3	0.106	0.007
UBPCN9	192000	0	3	0.108	0.006
UBPCN10	192000	0	3	0.113	0.008
UBPCN15	192000	0	4	0.154	0.014
UBPCN20	192000	0	4	0.229	0.027
UBPCN30	192000	0	5	0.274	0.039
IBED	192000	0	3	0.147	0.003

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UA	192000	0	0	0.15	0.004
IA	192000	0	0	0.167	0.007
IUA	192000	0	0	0.143	0.005
IAUA	192000	0	0	0.15	0.004
SVDSGD10I4F	192000	0	36	0.147	0.003
SVDSGD10I5F	192000	0	52	0.146	0.003
SVDSGD10I6F	192000	0	68	0.146	0.004
SVDSGD10I7F	192000	0	78	0.147	0.003
SVDSGD10I8F	192000	0	91	0.147	0.003
SVDSGD10I9F	192000	0	109	0.147	0.004
SVDSGD10I10F	192000	0	118	0.147	0.004
SVDSGD10I11F	192000	0	137	0.147	0.003
SVDSGD10I12F	192000	0	151	0.147	0.004
SVDSGD10I13F	192000	0	169	0.146	0.004
SVDALSWR10I1F	192000	0	4	0.142	0.004
SVDALSWR10I2F	192000	0	4	0.108	0.004
SVDALSWR10I3F	192000	0	5	0.083	0.003
SVDALSWR10I4F	192000	0	5	0.063	0.003
SVDALSWR10I5F	192000	0	6	0.043	0.002
SVDALSWR10I6F	192000	0	7	0.03	0.002
SVDALSWR10I7F	192000	0	8	0.023	0.001
SVDALSWR10I8F	192000	0	9	0.015	0.001
SVDALSWR10I9F	192000	0	11	0.008	0
SVDALSWR10I10F	192000	0	12	0.004	0
IBPC	192000	0	3	0.171	0.016
SO	192000	0	5	0.139	0.004

Notebooks WRMSE results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
NTB	192000	0	1	0.068	0.002
UBShdpmrN2	192000	0	12	0.032	0.002
UBShdpmrN3	192000	0	11	0.032	0.002
UBShdpmrN4	192000	0	12	0.033	0.002
UBShdpmrN5	192000	0	12	0.034	0.001
UBShdpmrN6	192000	0	13	0.035	0.001
UBShdpmrN7	192000	0	13	0.035	0.002
UBShdpmrN8	192000	0	14	0.036	0.001
UBShdpmrN9	192000	0	14	0.037	0.002
UBShdpmrN10	192000	0	14	0.039	0.001
UBShdpmrN15	192000	0	16	0.044	0.001
UBShdpmrN20	192000	0	18	0.049	0.002
UBShdpmrN30	192000	0	19	0.051	0.002
UBEDN2	192000	0	3	0.025	0.001
UBEDN3	192000	0	2	0.027	0.001
UBEDN4	192000	0	3	0.028	0.001
UBEDN5	192000	0	3	0.029	0.001
UBEDN6	192000	0	3	0.031	0.001
UBEDN7	192000	0	3	0.032	0.001
UBEDN8	192000	0	3	0.033	0.001
UBEDN9	192000	0	3	0.035	0.001
UBEDN10	192000	0	3	0.036	0.001
UBEDN15	192000	0	4	0.043	0.002
UBEDN20	192000	0	4	0.048	0.002
UBEDN30	192000	0	4	0.05	0.002
UBPCN2	192000	0	2	0.027	0.002
UBPCN3	192000	0	2	0.028	0.002
UBPCN4	192000	0	3	0.029	0.001
UBPCN5	192000	0	3	0.03	0.002
UBPCN6	192000	0	3	0.031	0.002
UBPCN7	192000	0	3	0.031	0.002
UBPCN8	192000	0	3	0.033	0.002
UBPCN9	192000	0	3	0.034	0.003
UBPCN10	192000	0	3	0.035	0.003
UBPCN15	192000	0	4	0.05	0.005
UBPCN20	192000	0	4	0.073	0.009
UBPCN30	192000	0	5	0.085	0.014
IBED	192000	0	3	0.044	0.001

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UA	192000	0	0	0.045	0.001
IA	192000	0	0	0.052	0.002
IUA	192000	0	0	0.044	0.002
IAUA	192000	0	0	0.046	0.001
SVDSGD10I4F	192000	0	35	0.043	0.001
SVDSGD10I5F	192000	0	50	0.043	0.001
SVDSGD10I6F	192000	0	62	0.043	0.001
SVDSGD10I7F	192000	0	77	0.043	0.001
SVDSGD10I8F	192000	0	93	0.043	0.001
SVDSGD10I9F	192000	0	109	0.043	0.001
SVDSGD10I10F	192000	0	124	0.043	0.001
SVDSGD10I11F	192000	0	134	0.043	0.001
SVDSGD10I12F	192000	0	146	0.043	0.001
SVDSGD10I13F	192000	0	162	0.043	0.001
SVDALSWR10I1F	192000	0	4	0.044	0.002
SVDALSWR10I2F	192000	0	4	0.033	0.001
SVDALSWR10I3F	192000	0	4	0.025	0.001
SVDALSWR10I4F	192000	0	5	0.019	0.001
SVDALSWR10I5F	192000	0	6	0.013	0.001
SVDALSWR10I6F	192000	0	6	0.009	0.001
SVDALSWR10I7F	192000	0	7	0.007	0
SVDALSWR10I8F	192000	0	8	0.005	0
SVDALSWR10I9F	192000	0	10	0.003	0
SVDALSWR10I10F	192000	0	11	0.001	0
IBPC	192000	0	3	0.051	0.005
SO	192000	0	5	0.043	0.001

Notebooks Tau results:

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
NTB	192000	0	1	0.251	0.01
UBShdpmrN2	192000	0	12	0.104	0.006
UBShdpmrN3	192000	0	12	0.105	0.006
UBShdpmrN4	192000	0	12	0.105	0.004
UBShdpmrN5	192000	0	13	0.109	0.005
UBShdpmrN6	192000	0	12	0.111	0.005
UBShdpmrN7	192000	0	13	0.114	0.004
UBShdpmrN8	192000	0	13	0.116	0.005
UBShdpmrN9	192000	0	14	0.12	0.004
UBShdpmrN10	192000	0	15	0.124	0.004
UBShdpmrN15	192000	0	19	0.14	0.004
UBShdpmrN20	192000	0	21	0.156	0.005
UBShdpmrN30	192000	0	22	0.164	0.005
UBEDN2	192000	0	3	0.083	0.003
UBEDN3	192000	0	3	0.085	0.003
UBEDN4	192000	0	3	0.089	0.003
UBEDN5	192000	0	3	0.093	0.003
UBEDN6	192000	0	3	0.097	0.002
UBEDN7	192000	0	3	0.101	0.003
UBEDN8	192000	0	3	0.105	0.003
UBEDN9	192000	0	3	0.11	0.003
UBEDN10	192000	0	3	0.114	0.003
UBEDN15	192000	0	4	0.135	0.004
UBEDN20	192000	0	4	0.153	0.004
UBEDN30	192000	0	5	0.162	0.006
UBPCN2	192000	0	3	0.089	0.005
UBPCN3	192000	0	3	0.091	0.005
UBPCN4	192000	0	3	0.094	0.004
UBPCN5	192000	0	4	0.096	0.005
UBPCN6	192000	0	3	0.099	0.004
UBPCN7	192000	0	4	0.102	0.005
UBPCN8	192000	0	4	0.105	0.005
UBPCN9	192000	0	4	0.11	0.01
UBPCN10	192000	0	4	0.115	0.01
UBPCN15	192000	0	4	0.157	0.014
UBPCN20	192000	0	4	0.231	0.031
UBPCN30	192000	0	4	0.272	0.041
IBED	192000	0	3	0.146	0.004

Builder	Est. cases	Non-Est. cases	Time[s]	Avg. Score	Deviation
UA	192000	0	0	0.15	0.004
IA	192000	0	0	0.168	0.006
IUA	192000	0	0	0.144	0.004
IAUA	192000	0	0	0.149	0.004
SVDSGD10I4F	192000	0	34	0.147	0.004
SVDSGD10I5F	192000	0	47	0.147	0.004
SVDSGD10I6F	192000	0	62	0.147	0.004
SVDSGD10I7F	192000	0	76	0.147	0.004
SVDSGD10I8F	192000	0	90	0.147	0.004
SVDSGD10I9F	192000	0	103	0.147	0.004
SVDSGD10I10F	192000	0	118	0.146	0.003
SVDSGD10I11F	192000	0	134	0.147	0.004
SVDSGD10I12F	192000	0	149	0.146	0.003
SVDSGD10I13F	192000	0	165	0.147	0.004
SVDALSWR10I1F	192000	0	4	0.141	0.004
SVDALSWR10I2F	192000	0	4	0.108	0.004
SVDALSWR10I3F	192000	0	4	0.083	0.003
SVDALSWR10I4F	192000	0	5	0.063	0.003
SVDALSWR10I5F	192000	0	6	0.043	0.002
SVDALSWR10I6F	192000	0	8	0.03	0.002
SVDALSWR10I7F	192000	0	9	0.023	0.001
SVDALSWR10I8F	192000	0	11	0.015	0.001
SVDALSWR10I9F	192000	0	12	0.008	0
SVDALSWR10I10F	192000	0	14	0.004	0
IBPC	192000	0	3	0.402	0.04
SO	192000	0	5	0.249	0.049