

Adaptive Systems

Practical Assignment 2: K-NN vs. SVD (Funk Variant)

Fabbri Lucia, Marza Mathilde

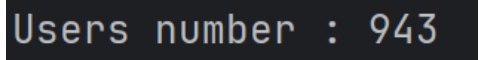
November 2023

1. Given this **data set** and the algorithm of K-NN explained in class for user-based CF

The dataset we're working with is named "MovieLens" and consists of 100.000 ratings (1-5) from 943 users on 1682 movies, collected by the GroupLens Research Project at the University of Minnesota. Each user has rated at least 20 movies and for each of them we have simple demographic info such as age, gender, occupation and zip.

To complete the two tasks a) and b) mentioned as follows, we decided to keep the same parameters given in the code examples: $n=5$ as the number of recommendations generated, $threshold=4$ as the minimum recommendation value to consider. In other words, this means that the performance of the model we're going to build is evaluated by looking at the 5 best recommendations it has generated, with a minimum level (threshold) to pass in order to be accepted as positive. Therefore, a recommendation for a specific user is considered relevant if the value is a 4 or a 5.

To find the best k for the KNN algorithm, in both cases we decided to test all the possible values, despite the computational cost spent to run the code. To do this, we looked for extreme values starting from 1 to the total number of users. As a result in the following Figure 1, the total number of users turns out to be 943 (as expected from the datasets information) and we therefore tested k values from 1 to 943.



Users number : 943

Figure 1: Total number of Users

a) Find out the value for K that minimizes the MAE with 25% of missing ratings.

Setting all the parameters as just mentioned above, we're now looking for the K-value that minimizes the MAE (Mean Absolute Error) with a 25% of missing rating, which refer to the missing user ratings in the dataset and it appears when user ratings for certain items that are not provided. These missing values can be a problem when building a recommendation model, since the model needs to make predictions for items that users haven't rated yet. The MAE is also another important metric because it represents the mean of the absolute differences between predicted and actual values. MAE range value goes from 0 to infinity and a lower MAE indicates higher model accuracy.

When determining the optimal value for K in KNN with 25% missing ratings, we are searching for the best number of neighbors K to consider for estimating the missing ratings in order to minimize the MAE. The results we have obtained are shown in the following Figure 2: the optimal value of K for minimizing the MAE is 75 (number of neighbors to consider) and the achieved lowest MAE is almost 0,745.

```

Sparsity: 25%
Best K for minimizing MAE: 75
Lowest MAE: 0.7446623578666356
  
```

Figure 2: KNN results with sparsity = 25%

To better understand the MAE trends considering the different number of K, we can plot a simple graph as shown in Figure 3, having on the x-axes the K-values and on the y-axes the different MAE values. As a result, we can easily notice that as k increases, the MAE decreases up to a point where it stagnates. We choose the lowest k for the lowest MAE in order to keep the model as simple as possible, which is 75, as highlighted with the red point in the graph.

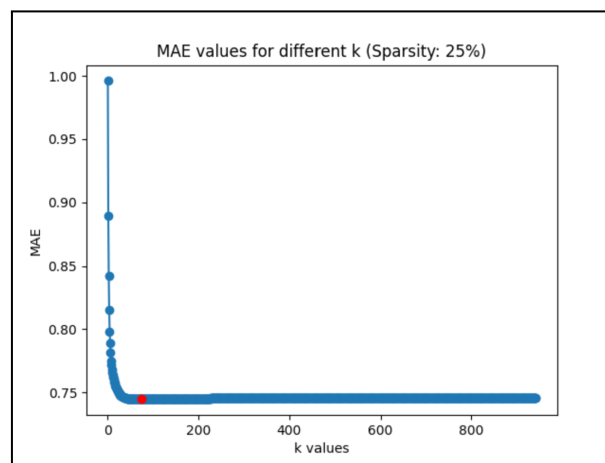


Figure 3: KNN results with sparsity = 25% for each k

b) Sparsity Problem: find out the value for K that minimizes the MAE with 75% of missing ratings.

The Sparsity Problem refers to the situation where the matrix user-item matrix has a relevant number of missing values (many users may haven't rated items) and this leads to a challenging situation for the KNN method to find a sufficient number of similar users/items for making accurate predictions.

Considering the same situation and parameters as before, as a requirement of task b), we have now raised the missing rating up to 75%: this means that the user/item matrix has now a higher number of missing values and, as a consequence, its density is lower. The following Figure 4 shows the results we have obtained performing this task with the new parameter settings: the optimal value of K for minimizing the MAE is 61 (number of neighbors to consider) and the lowest MAE we achieved is almost 0,813.

In this case, the MAE value obtained is quite higher compared to the one obtained in the previous task, with only 25% of missing ratings instead of 75%. The results seem to be logical since the

sparsity level is higher, i.e. more data is missing from the matrix, making it more difficult to predict relevant films for KNN.

```
Sparsity: 75%
Best K for minimizing MAE: 61
Lowest MAE: 0.8125340341452447
```

Figure 4: KNN results with sparsity = 75%

As previously described in Figure 3, the following Figure 5 describes the MAE trend when considering a sparsity level of 75%. Also in this case, we have on the x-axes the K-values and on the y-axes the different MAE values and we can easily notice that as k increases, the MAE decreases up to a point where it stagnates. We choose the lowest k for the lowest MAE in order to keep the model as simple as possible, which is 61, as highlighted with the red point in the graph.

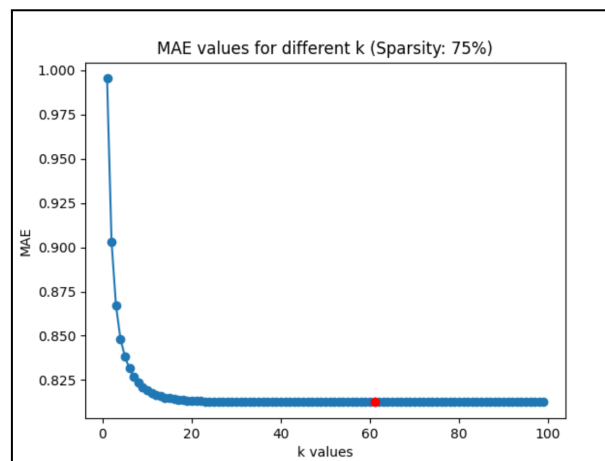


Figure 5: KNN results with sparsity = 25% for each k

2. Mitigation of sparsity problem: show how SVD (Funk variant) can provide a better MAE than user-based K-NN using the provided data set.

SVD (Singular Value Decomposition) is a technique used in recommendation systems that decomposes the user-item matrix into 3 simpler terms/matrices: one for users, one for items and one for singular values. In particular, the Funk variant uses the stochastic gradient descent approach, allowing the model to adapt and learn patterns particularly in scenarios with sparse datasets. This is the main reason why SVD Funk Variant is so common and used in addressing Sparsity Problems as mentioned in the previous paragraphs.

Performing the same task as we did for the KNN methods and still using the two different missing ratings of 25% and 75%, we noticed that the results were changed. As expected, we noticed that considering the same level of sparsity, in both cases the MAE value reached is lower compared to

the one obtained using KNN. In fact, the SVD Funk variant provides better results in mitigating the sparsity problem due to its global approach, dimensionality reduction capabilities, adaptability to sparse data, and robustness to noise.

The following Figure 6 shows that considering a sparsity level of 25%, the achieved MAE is almost 0,739, while in Figure 7 we can notice that with a sparsity level of 75%, the achieved MAE reaches almost 0,774: in both cases the results are slightly better for SVD than for KNN. However, with a low sparsity level of 25%, KNN and SVD are almost equivalent, but this latter turns out to be much more effective than KNN with a high sparsity level of 75%.

Sparsity: 25%
MAE: 0.7387569722703375

Figure 6: SVD results with sparsity = 25%

Sparsity: 75%
MAE: 0.7735654307074209

Figure 7: SVD results with sparsity = 75%

Results summary table

In the following Table 1, we list all the result we have obtained while applying to both KNN and SVD methods to our datasets and evaluate the different performances using the MAE metrics with 25% and 75% of missing ratings:

	KNN	SVD
M.R. 25%	0,745	0,739
M.R. 75%	0,813	0,774

Table 1: Summary table with all the results obtained

To sum up, KNN is a method based on local similarity and can struggle with more missing data, leading to increased errors. In contrast, SVD shows better resistance to high sparsity, consistently keeping MAE values lower.

3. Top-N recommendations: calculate the precision, recall, and F1 with different values for N (10..100) using user-based K-NN (with the best Ks) and SVD. To do this, you must suppose that the relevant recommendations for a specific user are those rated with 4 or 5 stars in the data set. Perform the calculations for both 25% and 75% of missing ratings.

Once having tuned our model considering the parameters given by the previous task, it's now important to evaluate our models using specific metrics, such as precision, recall and F1-score. Before showing the results we have obtained, it's important to give an introduction and an overview of what these metrics mean and why they're so important.

- *Precision*: it's a metric that assesses the accuracy of positive predictions of the model, by focusing on minimizing false positives, which occur when the model incorrectly identifies instances as positive when they are actually negative. To be precise, it corresponds to how many of the predicted positive instances are actually positive.

Precision is calculated as the ratio of true positives to the sum of true positives and false positives, as shown in the following formula: a precision of 1 indicates that all instances predicted as positive are positive, while 0 indicates that no positive predictions are correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- *Recall*: also known as sensitivity or true positive rate (TPR), it's a metric that measures the ability of a model to correctly identify all relevant instances of the positive class. It is particularly concerned with minimizing false negatives, which occur when the model fails to identify positive instances that are actually positive. To be precise, it corresponds to how many of the actual positive instances are successfully identified by the model.

Recall is calculated as the ratio of true positives to the sum of true positives and false negatives, as shown in the following formula: a recall of 1 indicates that the model correctly identified all the positive instances, while 0 indicates that no positive instances were correctly identified.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- *F1-score*: it's a metric especially useful when there is a need to find a compromise between precision and recall, because it is easy to have very high recall if precision decreases, just as it is easy to have very high precision if recall decreases. So, the goal is not to have one of both very high and the other one down. It combines precision and recall into a single value, providing a balanced assessment of a model's performance. For this reason, it is commonly

used in scenarios where both false positives and false negatives are critical. F1-score ranges from 0 to 1, where a higher score indicates a better balance between precision and recall. F1-score is calculated as the harmonic mean of precision and recall, as shown in the following formula:

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

Considering the importance of the metrics shown so far, we're now presenting on separate graphs (Figure 8, Figure 9, Figure 10 and Figure 11) all the results we obtained for KNN and SVD Funk Variant, with a sparsity level of 25% and 75%. The results cover a range of values for every n from 10 to 100, incorporating the optimal K-value identified in the previous sections.

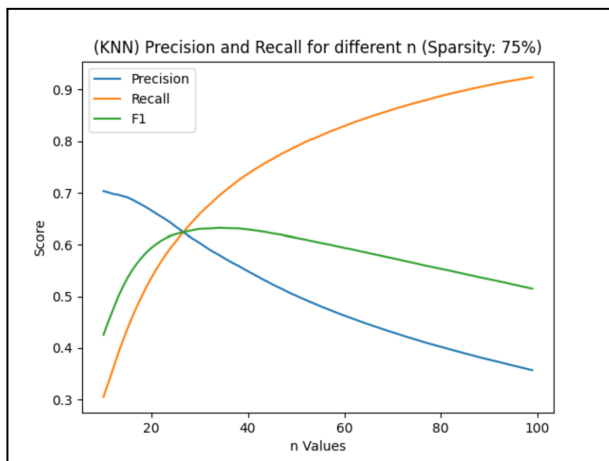


Figure 8: KNN pre/rec/F1, sparsity=25%, k=75

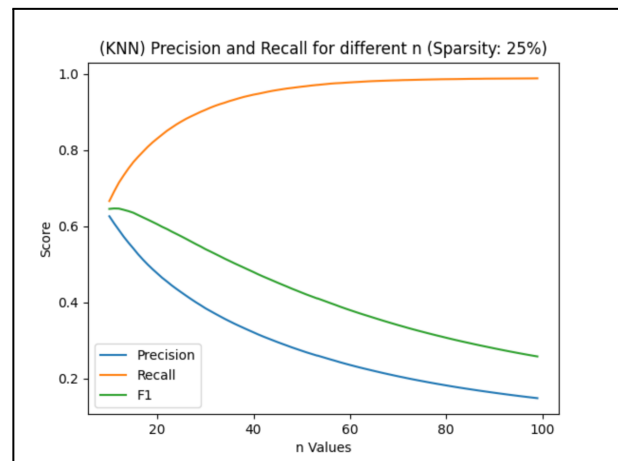


Figure 9: KNN pre/rec/F1, sparsity=25%, k=61

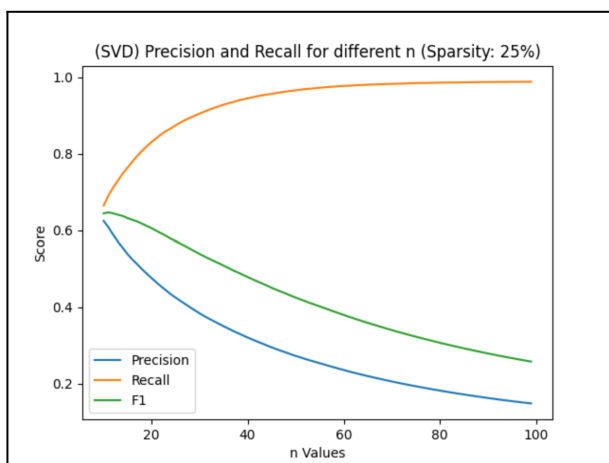


Figure 10: SVD pre/rec/F1, sparsity=25%

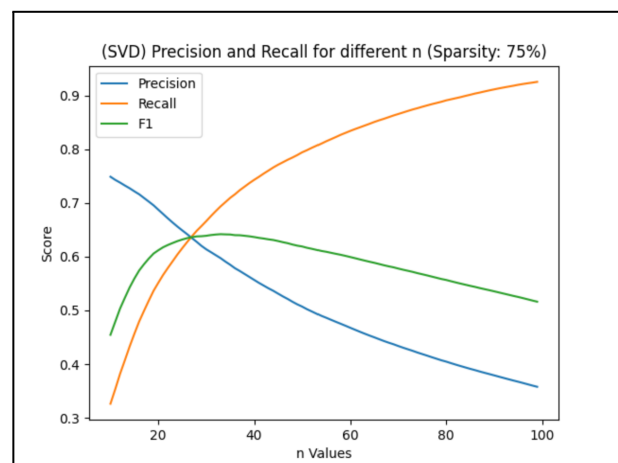


Figure 11: SVD pre/rec/F1, sparsity=75%

As we would expect, increasing the value of n , we can notice that the recall (the ability to find all relevant items) goes up, but precision (the accuracy of positive predictions) and the F1 score (a

balance between precision and recall) go down for both KNN and SVD at 25% and 75% sparsity levels: this is because a higher n suggests more different recommendations. As mentioned before, recall is the ability to catch all the movies we'd really like (true positives). When n is larger, we're more likely to catch all the good ones. However, precision decreases because it becomes trickier to ensure that everything recommended is actually relevant. So, the chances that what's predicted as positive is really positive decrease. In simpler terms, having more recommendations can help us not miss out on good movies (higher recall), but it also increases the chance of suggesting some that aren't as relevant (lower precision). Balancing these aspects is crucial, and that's what the F1 score helps us evaluate.

Going more into details, for each method and sparsity level, we're now analyzing and comparing the various graphs with each other.

a) KNN method

As far as we know, the KNN method encounters challenges in handling data sparsity effectively. This challenge is evident in Figure 12, where the recall is significantly higher for KNN at 25% sparsity compared to KNN at 75%. The reason behind lies in the fact that while increasing n , we also enhance the chances of identifying all the positive films, while the model's predictive capability differs between KNN 25% and KNN 75%.

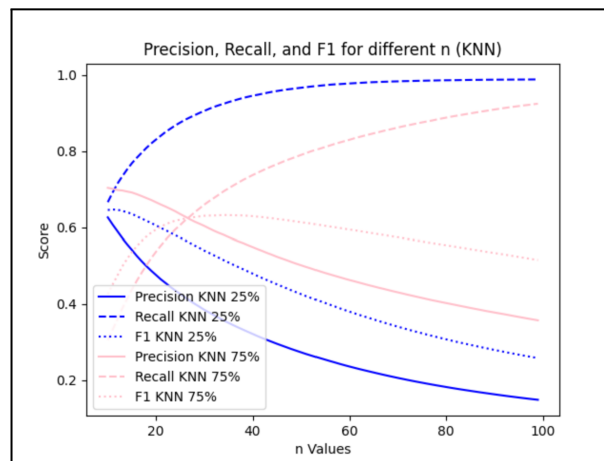


Figure 12: KNN graph's results

For KNN 25%, the model seems to make more accurate predictions, resulting in a higher recall and suggesting that it's better at capturing positive instances. On the other hand, KNN 75% faces difficulties when dealing with too many empty cells in the sparse data and it may not make as many relevant predictions for positive cases, leading to a lower recall.

Concerning the precision curve, we can observe that the value is higher for KNN 75% compared to KNN 25%. We found out that this result could be attributed to different factors, such as the algorithm's adaptability to handling sparse data, the effectiveness of local pattern recognition,

variations in the optimal k-value and the intrinsic characteristics of the specific sparse dataset. All of this emphasizes the importance of considering both precision and recall for a comprehensive evaluation, and so F1-score.

b) SVD Funk Variant method

Because SVD handles sparse data better than KNN, the below Figure 13 shows similar trends as we observed in the comparison between KNN at 75% and 25% sparsity. However, Sparsity 75% performs much better with SVD than with KNN 75%. In simpler terms, the pink curves show better results for all three metrics.

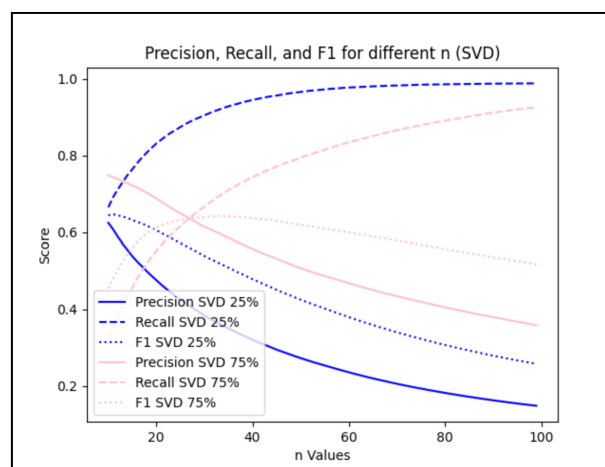


Figure 13: SVD graph's results

c) KNN and SVD Funk Variant, with sparsity level = 25%

In situations with low missing ratings (25% sparsity), Figure 14 illustrates overlapping curves, indicating similar performance between KNN and SVD Funk Variant. This suggests that both methods can offer meaningful recommendations when data is limited.

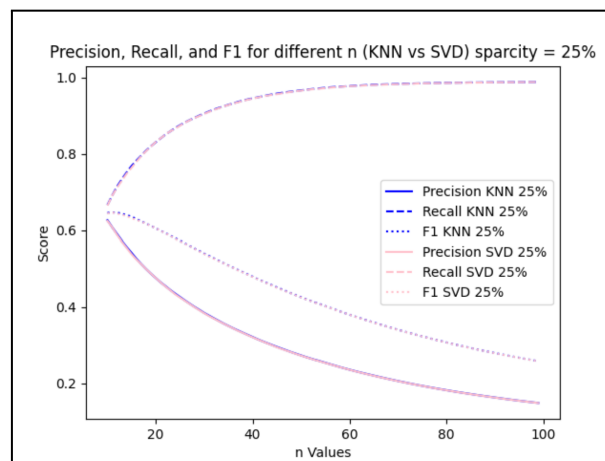


Figure 14: KNN and SVD graph's results, sparsity=25%

d) KNN and SVD Funk Variant, with sparsity level = 75%

In scenarios with high sparsity (75% missing ratings), Figure 15 shows curves that overlap, indicating that both KNN and SVD Funk Variant perform similarly. However, a closer look reveals a slight difference in the trend of the curves, with SVD maintaining a small advantage. SVD's ability to handle high sparsity conditions slightly better than KNN, could be contributing to its edge in performance.

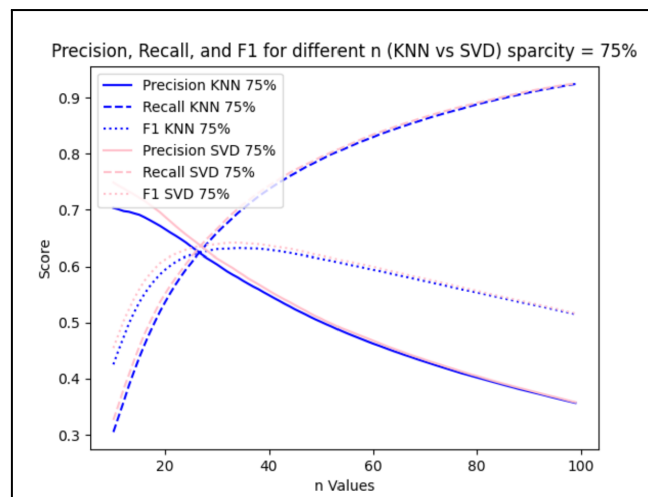


Figure 15: KNN and SVD graph's results, sparsity=75%

Conclusion

In conclusion, our dive into how KNN and SVD Funk Variant perform with the MovieLens dataset revealed some interesting findings. When the data had a lot of missing ratings (75% sparsity), KNN faced more challenges, while SVD Funk Variant handled the sparsity issue quite well.

We mainly looked at three metrics: Precision, Recall, and F1-score, which focus on how well the models predict positive cases. However, it would be intriguing to consider negative cases too. Metrics like specificity (TNR) that reflect the recall for negative instances could provide a more complete picture. Looking at ROC curves, which show how well models distinguish between true and false positives at different decision thresholds, and PR curves, which combine recall and precision across various decision thresholds, could give us even more insights.

In simple terms, by exploring both positive and negative aspects and using different metrics, we get a better understanding of how these models perform. This approach helps us fine-tune and improve both KNN and SVD Funk Variant for real-world scenarios where recommendation systems are applied.