CHAPTER 4
ANALYSIS AND DESIGN

4.1 Pre Processing Dataset

In this study, the author uses two algorithms, namely collaborative filtering item-based using cosine similarity and also nave Bayes in comparing the two algorithms in the film recommendation engine. Where the dataset used is the same for both algorithms, namely the dataset that the author took from MovieLens with references from Kaggle. In this dataset, there are approximately 105,000 data provided but what the author will use is 10,000 datasets in this study. In the dataset, there are several data sheets that have been provided by MovieLens, but the author only took 2 sheets from several sheets that have been provided, namely the movie sheet and also the rating sheet.

4.1.1 Table Rating Dataset

userId	movieId	Rating	timestamp
1	3	1.4	1.22E+09
1	9	3.5	1.22E+09
2	3	4.6	1.22E+09
3	7	4.3	1.22E+09
3	3	2.7	1.22E+09
4	10	4.8	1.22E+09
4	7	2.5	1.22E+09
4	10	3.6	1.22E+09
5	3	3.4	1.22E+09
5	6	2.7	1.22E+09

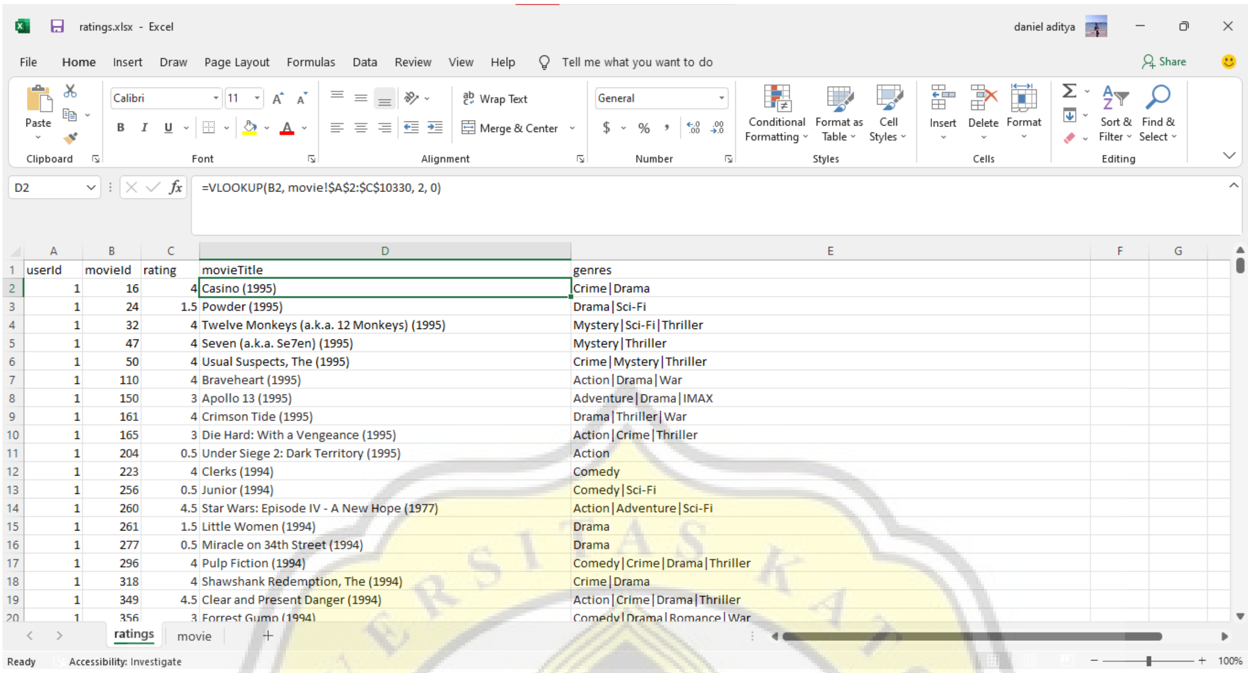
4.1.2 Table Movie Dataset

movieId	title	Genres
1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

2	Jumanji (1995)	Adventure|Children|Fantasy
3	Grumpier Old Men (1995)	Comedy|Romance
4	Waiting to Exhale (1995)	Comedy|Drama|Romance
5	Father of the Bride Part II (1995)	Comedy
6	Heat (1995)	Action|Crime|Thriller
7	Sabrina (1995)	Comedy|Romance
8	Tom and Huck (1995)	Adventure|Children
9	Sudden Death (1995)	Action
10	GoldenEye (1995)	Action|Adventure|Thriller

It can be seen in the rating dataset table (4.1.1) that there are various columns such as userId, movieId which is a foreign key of movieId in the movie dataset table (4.1.2), rating, and also timestamp. Then in the movie dataset table (4.1.2), there is a movieId which is the primary key in the movie dataset table, a title that contains the title of the movie, and also genres which contains the genre category of the movie in the movieId. In preprocessing this data, the author combines the two sheets into one with Microsoft Excel tools using the vLookUp feature by combining the two sheets into one first. Then the author omitted the timestamp column in the rating dataset because it was deemed not to be used. Then the author uses the vlookup function so

4.1.3 Function vLookUp Microsoft Excel



where B2 is column B in the second row which contains movieId 16. Then the author takes the reference data that we will vlookup from the movie which can be seen by the author using the AC column in the vlookup in the movie sheet where column A is movieId, B is movieTitle and C is genres his. To get the movieTitle, the author takes the second value as can be seen at the end of the formula, which is 2 and the number 0 at the end of the formula is the comparison we use to find the same data between sheets where the data this time is movieId. So that the author gets the final data results like

4.1.4 Table ratingMovie

userId	movieId	rating	movieTitle	genres
1	3	1.4	Grumpier Old Men (1995)	Comedy|Romance
1	9	3.5	Sudden Death (1995)	Action
2	3	4.6	Grumpier Old Men (1995)	Comedy|Romance
3	7	4.3	Sabrina (1995)	Comedy|Romance
3	3	2.7	Grumpier Old Men (1995)	Comedy|Romance
4	10	4.8	GoldenEye (1995)	Action|Adventure|Thriller

4	7	2.5	Sabrina (1995)	Comedy|Romance
4	10	3.6	GoldenEye (1995)	Action|Adventure|Thriller
5	3	3.4	Grumpier Old Men (1995)	Comedy|Romance
5	6	2.7	Heat (1995)	Action|Crime|Thriller

Where the author only uses a few columns such as userId, movieId, rating, movieTitle, and also genres.

Then for later use in calculating the MSE and RMSE formulas, the author has prepared data for the actual data, namely by processing the data table (4.1.1 Rating Dataset) by calculating the average rating and also sorting the data based on the highest average rating and also based on the number most ratings. First, the writer enters the two data from (tables 4.1.1 and 4.1.2) into the database which becomes 2 tables then makes a query from the two tables which becomes the movie ranking data from this dataset.

4.1.5 Query untuk mendapatkan ranking movie data

```
SELECT DISTINCT(movie.movieId), sum(rating) as total_rating, count(rating) jumlah_rating, round(sum(rating)/count(rating), 3) as average_rating , movie_title.title as title, movie_title.genre as genres
FROM `movie`
left join movie_title on movie.movieId = movie_title.movieId
GROUP BY(movieId)
order by average_rating desc, sum(rating) desc
```

4.1.6 Ranking Dataset

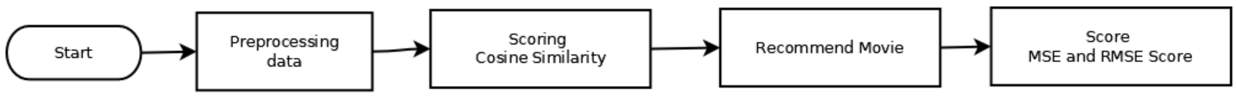
movieId	averageRating	movieTitle	genres
10	4.2	GoldenEye (1995)	Action|Adventure|Thriller

9	3.5	Sudden Death (1995)	Action
7	3.4	Sabrina (1995)	Comedy|Romance
4	3.4	Waiting to Exhale (1995)	Comedy|Drama|Romance
8	3.1	Tom and Huck (1995)	Adventure|Children
3	3.025	Grumpier Old Men (1995)	Comedy|Romance
6	2.7	Heat (1995)	Action|Crime|Thriller
1	2.4	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	1.3	Jumanji (1995)	Adventure|Children|Fantasy
5	1.1	Father of the Bride Part II (1995)	Comedy

4.2 Collaborative Filtering

In this research, collaborative filtering used by the author is item-based collaborative filtering using the cosine similarity formula. There are several steps contained in collaborative filtering that will be used by the author.

4.2.1 Flowchart collaborative Filtering



First, the author will preprocess the data from the ratingMovie table (table 4.1.4) again so that we can use the formula for this cosine similarity later. Where in this preprocessing we will group the rating ratings in 1 column based on the same movieId and eliminate the userId column, and also we will combine these ratings with a comma separator (,) which aims to be able to separate them later using the split function in python. So from the ratingMovie dataset above, the writer gets the preprocessing data as follows.

4.2.2 Table Preprocessing Data Collaborative Filtering

movieId	rating	movieTitle	genres
3	1.4, 4.6, 2.7, 3.4	Grumpier Old Men (1995)	Comedy|Romance
9	3.5	Sudden Death (1995)	Action
7	4.3, 2.5	Sabrina (1995)	Comedy|Romance
10	4.8, 3.6	GoldenEye (1995)	Action|Adventure|Thriller
6	2.7	Heat (1995)	Action|Crime|Thriller

After preprocessing the data, the author makes input for the user or the author himself to determine the movie reference that will be the reference item by selecting the movieId. In collaborative filtering with cosine similarity, this time the cosine similarity formula used by the author is

4.2.3 Function Cosine Similarity

$$\cos\theta = \frac{A.B}{||A||.||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

where A is obtained from the movieId input by the user or author, and B is all data except the data A itself. To calculate the formula there are several steps. The first is to calculate A.B , in this example, the author will assume that A is movieId 3 and B is 9.

$$A.B = \sum_{i=1}^{n} A_i B_i = (1.4 * 4.6 * 2.7 * 3.4) + (3.5) = 62.6$$

For ||A|| and also ||B|| will look like this.

$$||A|| = \sqrt{1.4^2 + 4.6^2 + 2.7^2 + 3.4^2} = 6.478$$

$$||B|| = \sqrt{3.5^2} = 3.5$$

So when combined, the cosine similarity value of movieId 9 to movieId 3 is

$$\cos\theta = \frac{62.6}{6.478 * 3.5} = 2.76$$

After the author conducted testing on all movieIds against movieId 3, the following results were obtained (Cosine similarity score results were taken from all movieIds against movieId 3).

4.2.4 Table Cosine Similarity Score

movieId	rating	movieTitle	score	genres
3	1.4, 4.6, 2.7, 3.4	Grumpier Old Men (1995)	-	Comedy|Romance
9	3.5	Sudden Death (1995)	2.76	Action
7	4.3, 2.5	Sabrina (1995)	2.168	Comedy|Romance
10	4.8, 3.6	GoldenEye (1995)	1.965	Action|Adventure|Thriller
6	2.7	Heat (1995)	3.533	Action|Crime|Thriller

4.2.5 Table Cosine Similarity Sorted

movieId	rating	movieTitle	score	genres

6	2.7	Heat (1995)	3.533	Action|Crime|Thriller
9	3.5	Sudden Death (1995)	2.76	Action
7	4.3, 2.5	Sabrina (1995)	2.168	Comedy|Romance
10	4.8, 3.6	GoldenEye (1995)	1.965	Action|Adventure|Thriller
3	1.4, 4.6, 2.7, 3.4	Grumpier Old Men (1995)	-	Comedy|Romance

After getting the value from the cosine similarity, the writer displays some of the results of the sequence of movie titles and also their genres based on the top order of the score.

Then as a comparison value, the author calculates the MSE and RMSE from the above data. By using the following formula:

4.2.6 Function MSE

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (fi - yi)^2$$

where n is the number of data entries, which is 5 in this example based on (Table 4.2.5), then Fi is the actual output which here we assume as the average rating of each movie from the movie ranking in (Table 4.1.6) , and Yi is the predicted output, which is the rating based on the score from the calculation of cosine similarity.

$$MSE = \frac{1}{5} * \ (2.7 - 3.533)^2 + (3.5 - 2.76)^2 + (3.4 - 2.168)^2 + (4.2 - 1.965)^2 = 1.55$$

$$MSE = \frac{1}{10} * \ ((2.7 - 4.2)^2 + (3.5 - 3.5)^2 + (3.4 - 3.4)^2 + (4.2 - 3.4)^2 + (3.025 - 3.1)^2) = 2.89005625$$

Then for the calculation of RMSE, the calculation used with MSE is actually almost the same, only the difference is that RMSE uses the root for the final result.

4.2.7 Function RMSE

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (fi - yi)^2}$$

$$RMSE = \sqrt{\frac{1}{10} * ((2.7 - 4.2)^2 + (3.5 - 3.5)^2 + (3.4 - 3.4)^2 + (4.2 - 3.4)^2 + (3.025 - 3.1)^2)} = 0.289005625$$

And the MSE result is 2.89005625 while the RMSE is 0.289005625.

4.3 Naïve Bayes

In making a movie recommender using the nave Bayes formula, this time the author uses the following plot.

In this study, the author started by preprocessing the data that was already owned from the results of preprocessing the previous data (Table 4.1.4). In preprocessing data, here we will try to regroup the rating data for each movie that has been given by the user based on its userId so that we have preprocessing data like the following. And also the author groups the rating data given by the user to a movie to be able to add up how many ratings the user has given to the existing movies.

4.3.2 Movie Data

movieId	Rating [userId, rating]	averageRating
3	[1, 1.4],[2, 4.6], [3, 2.7], [5, 3.4]	3.025
9	[1, 3.5]	3.5
7	[3, 4.3], [4, 2.5]	3.4
10	[4, 4.8], [4, 3.6]	4.2
6	[5, 2.7]	2.7

4.3.3 Table User Data

userId	Rating [movieId, rating]
1	[3, 1.4],[9, 3.5]
2	[3, 4.6]
3	[7, 4.3], [3, 2.7]
4	[10, 4.8], [7, 2.5], [10, 3.6]
5	[3, 3.4], [6, 2.7]

The author groups all rating values on the same movieId as the separator by providing userId information on the rating given to the movieId, then there are also authors calculating the movieId collection that has been rated by the user by making it a separate array. After getting the values in the table above, we begin to perform calculations using the nave Bayes formula, namely the calculation of probabilities with the following formula:

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

4.3.3 Function Naïve Bayes

To be able to use the nave Bayes formula, we will calculate the probability of movie A against all the probabilities of the rating from the user that has been given to the movie A. So P(B|A) is the rating value divided by all the ratings for that 1 movie, then P(A) is the number of ratings on 1 movie divided by the number of all ratings on all the movies, then P(B) is the number of rating values given by the user divided by the number of all ratings that the user has given to all existing movies divided by the number of all existing ratings. For example, we will calculate the probability of movieId 3 then:

$$P(B|A) = \frac{\frac{1.4}{12.1} * \frac{4.6}{12.1} * \frac{2.7}{12.1} * \frac{3.4}{12.1} * \frac{12.1}{33.5}}{\frac{4.9}{33.5} * \frac{4.6}{33.5} * \frac{7}{33.5} * \frac{6.1}{33.5}} = \frac{0.115 * 0.380 * 0.223 * 0.280 * 0.361}{0.146 * 0.137 * 0.208 * \ 0.182}$$

$$= \frac{9.85034708xe - 6}{0.000757195712} = 0.01301$$

movieId	score	averageRating
3	0.0130	3.025
9	0.7143	3.5
7	0.6074	3.4
10	0.5801	4.2
6	0.4426	2.7

After calculating all the nave Bayes scores from each movie to the user, the writer then sorts the data based on the nave Bayes scores.

movieId	score	averageRating
9	0.7143	3.5
7	0.6074	3.4
10	0.5801	4.2
6	0.4426	2.7
3	0.0130	3.025

Then after getting the order based on the highest nave Bayes score, the author provides movie recommendations according to the data that has been sorted. After that the author calculates the MSE value and RMSE value using the formula:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (fi - yi)^2$$

Where Fi is the sequential rating value of the top ranking based on the nave Bayes score, then Yi is the sequential rating value of the top ranking of the actual data or data that we have sorted (Table 4.1.6). Then the author calculates the total rating on the nave Bayes score with the rating in (table 4.1.6) which we then rank and multiply by 1/ the amount of data on the rating on the Nave Bayes score.

$$MSE = \frac{1}{10} * ((3.5 - 4.2)^2 + (3.4 - 3.5)^2 + (4.2 - 3.4)^2 + (2.7 - 3.4)^2 + (3.025 - 3.1)^2) = 0.1635625$$

Then to calculate the RMSE the author uses the following formula:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (fi - yi)^2}$$

$$RMSE = \sqrt{\frac{1}{10} * ((3.5 - 4.2)^2 + (3.4 - 3.5)^2 + (4.2 - 3.4)^2 + (2.7 - 3.4)^2 + (3.025 - 3.1)^2)} = 0.4044286093$$

4.4 Perbandingan Collaborative Filtering dengan Naïve Bayes

In this test, the author compares the two algorithms between collaborative filtering using cosine similarity with nave Bayes in giving recommendations to movies. The comparison is done by the author using MSE (Mean Square Error) and also RMSE (Root Mean Square Error), where MSE is the average squared error value between the original image and the predicted image, while RMSE itself is a measurement method by measuring the difference in the value of the prediction. a model as an estimate of the observed value.

The smaller the MSE and RMSE values, the better the algorithm. From the example above, we get:

	MSE	RMSE
Collaborative Filtering	1.55	2.89005625
Naïve Bayes	0.1635625	0.4044286093

From the data above, we can see that the best algorithm with the same dataset between collaborative filtering and nave Bayes is nave Bayes because nave Bayes has a smaller MSE value and RMSE value compared to collaborative filtering.