

|  |
| --- |
| Group 8 |

|  |
| --- |
|  |
| DTI5125: Data Science Applications |
| Final Project |

Table of Contents

[1. Problem Formulation and system overview: 4](#_Toc110395147)

[2. Dataset: 4](#_Toc110395148)

[3. Classification: 4](#_Toc110395149)

[a) Data Preparation: 4](#_Toc110395150)

[b) Feature Engineering: 5](#_Toc110395151)

[c) Models: 6](#_Toc110395152)

[d) Results: 6](#_Toc110395153)

[e) Error Analysis: 9](#_Toc110395154)

[4. Clustering: 9](#_Toc110395155)

[a) Data Preparation: 9](#_Toc110395156)

[b) Feature Engineering: 10](#_Toc110395157)

[c) Models: 10](#_Toc110395158)

[5. Recommender: 11](#_Toc110395159)

[a) Data Preparation: 11](#_Toc110395160)

[b) Building the recommender: 11](#_Toc110395161)

[1. Popularity Based (Top In the whole collection): 11](#_Toc110395162)

[2. Recommendation using Average Weighted Rating: 12](#_Toc110395163)

[3. User-Item Collaborative Filtering Recommendation: 12](#_Toc110395164)

[4. Recommendation using surprise library (specific to a user): 13](#_Toc110395165)

[5. Recommendation using surprise library (not specific to a user): 13](#_Toc110395166)

[6. Chatbot design: 14](#_Toc110395167)

[a) Intents design: 14](#_Toc110395168)

[b) Evaluation (overall system): 17](#_Toc110395169)

[c) Error Analysis (overall system): 18](#_Toc110395170)

[References: 19](#_Toc110395171)

Table of Figures:

[Figure 1 The overall sequence of the system 4](#_Toc110395172)

[Figure 2 The data set before preparation 4](#_Toc110395173)

[Figure 3 The dataset after removing extra columns 5](#_Toc110395174)

[Figure 4 The dataset after preparation 5](#_Toc110395175)

[Figure 5 Coherence of LDA features using different number of topics 5](#_Toc110395176)

[Figure 6 The frequency of top 30 words in topic 2 6](#_Toc110395177)

[Figure 7 Training and Validation Accuracies of the models 8](#_Toc110395178)

[Figure 8 The top 10 words in misclassified reviews 9](#_Toc110395179)

[Figure 9 The data set before preparation 9](#_Toc110395180)

[Figure 10 The dataset after removing extra columns 10](#_Toc110395181)

[Figure 11 Elbow method with K-means 10](#_Toc110395182)

[Figure 12 Number of ratings for each book 11](#_Toc110395183)

[Figure 13 The recommended books 12](#_Toc110395184)

[Figure 14 Recommended books using Average Weighted Rating 12](#_Toc110395185)

[Figure 15 The recommended books given "ERAGON: INHERITANCE, BOOK ONE." book as an input 12](#_Toc110395186)

[Figure 16 The recommended books given "The Devil Wears Prada" book as an input 13](#_Toc110395187)

[Figure 17 Example for user ID 13](#_Toc110395188)

[Figure 18 The five recommended books for this specific user 13](#_Toc110395189)

[Figure 19 The number of books vs the average rating 13](#_Toc110395190)

[Figure 20 Scatterplot shows the total number of ratings and the average rating for each book 14](#_Toc110395191)

[Figure 21 Passing the book name and "Positive" to get the book recommendation 14](#_Toc110395192)

[Figure 22 The recommended book 14](#_Toc110395193)

[Figure 23 Passing the book name and "Neutral" to get the book recommendation 14](#_Toc110395194)

[Figure 24 The recommended book 14](#_Toc110395195)

[Figure 25 The developed intents 15](#_Toc110395196)

[Figure 26 The fulfillment 17](#_Toc110395197)

[Figure 27 The book name and the user feedback that the “recommender followup” intent took and the classifier prediction of the feedback 18](#_Toc110395198)

[Figure 28 The response of the chatbot for the user feedback 19](#_Toc110395199)

# Problem Formulation and system overview:

Nowadays, there are many books in many different fields, making it difficult to locate a decent book that is appropriate for the reader. So, we use a chatbot to recommend a book based on collaborative filtering. First, the chatbot recommends a random book for the user. Secondly, the user responds with a book review. Then, the chatbot calls API using the URL provided by NGROK to access the API running in Google Colab to classify the user’s review into “Positive”, “Neutral”, or “Negative”. Lastly, using the book name, user review, history reviews of other users, we got the recommendation and return it back to the user as shown in Figure 1.

The classification part is responsible for classifying the reviews into three classes (“Positive”, “Neutral”, “Negative”). The clustering part is responsible for clustering similar books based their descriptions; it can be used with the chatbot as a future work. The recommender part proposes five different ways to recommend the books (only the last way used with the chatbot). Dialogflow is used to develop the chatbot.

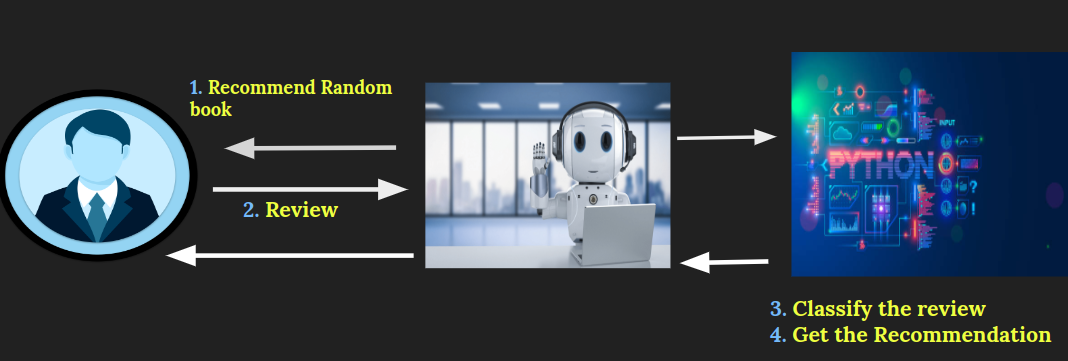
****

Figure 1 The overall sequence of the system

# Dataset:

The used dataset is book reviews from Amazon [1]. Google books API is used to get the descriptions of the books that are needed by clustering.

# Classification:

In this part, book reviews data from Amazon is used. The reviews of the books will be classified into three classes (“Positive”, “Neutral”, “Negative”) based on user rating. The steps are as follows:

## Data Preparation:

Graphical user interface, text, application

Description automatically generated

Figure 2 The data set before preparation

* 1. One million reviews are taken from the dataset.
  2. Removing extra columns. Seven columns out of eleven columns are removed.

Graphical user interface, application

Description automatically generated

Figure 3 The dataset after removing extra columns

* 1. Renaming columns.
  2. Removing the users that don’t have userID.
  3. Removing duplicated rows.
  4. Constructing the “class” column that have the class of the review (“Positive”, “Neutral”, “Negative”).
  5. Choosing balanced sample from the data (7000 row for each class).
  6. Cleaning the reviews text by removing the digits, extra spaces, and special characters.

Diagram

Description automatically generated with medium confidence

Figure 4 The dataset after preparation

## Feature Engineering:

We used two types of feature Engineering as follows:

* 1. TF-IDF.
  2. LDA. To determine the optimal number of topics, coherence measure is used to determine the optimal value for topic numbers.

Chart, line chart

Description automatically generated

Figure 5 Coherence of LDA features using different number of topics

As the above figure shows the best value is 70 topics.

Chart

Description automatically generated

Figure 6 The frequency of top 30 words in topic 2

We visualized the results as shown in the above figure to show the top 30 words in each topic.

## Models:

In this step we train multiple models (logistic regression, RandomForest Classifier, naive bayes) each model will be trained with each feature engineering techniques (TF-IDF and LDA).

## Results:

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Confusion Matrix on validation data | Train Accuracy | Validation Accuracy |
| TF\_IDF and logistic regression |  | 84.94% | 66.76% |
| TF\_IDF and RandomForest Classifier |  | 74.60% | 56.28% |
| TF\_IDF and naive bayes |  | 83.01% | 61.09% |
| LDA and logistic regression |  | 43.98% | 43.90% |
| LDA and RandomForest Classifier |  | 67.69% | 43.71% |
| LDA and naive bayes |  | 43.73% | 43.47% |

Chart, bar chart

Description automatically generated

Figure 7 Training and Validation Accuracies of the models

From the above figure the champion model will be Logistic Regression using TF-IDF features (highest validation accuracy).

## Error Analysis:

Chart, bar chart

Description automatically generated

Figure 8 The top 10 words in misclassified reviews

From the above figure, it is concluded that the most frequent words in the misclassified reviews are stop words. So, it seems that removing these words from the beginning may increase the accuracy of the model.

|  |  |
| --- | --- |
|  |  |

The above figures show the word clouds of two misclassified reviews (“this” is common between them).

# Clustering:

In this part, book description data that is got using Google Book API is used. The books are clustered based on these descriptions. The steps are as follows:

## Data Preparation:

* 1. The Descriptions of ten thousand books are acquired.

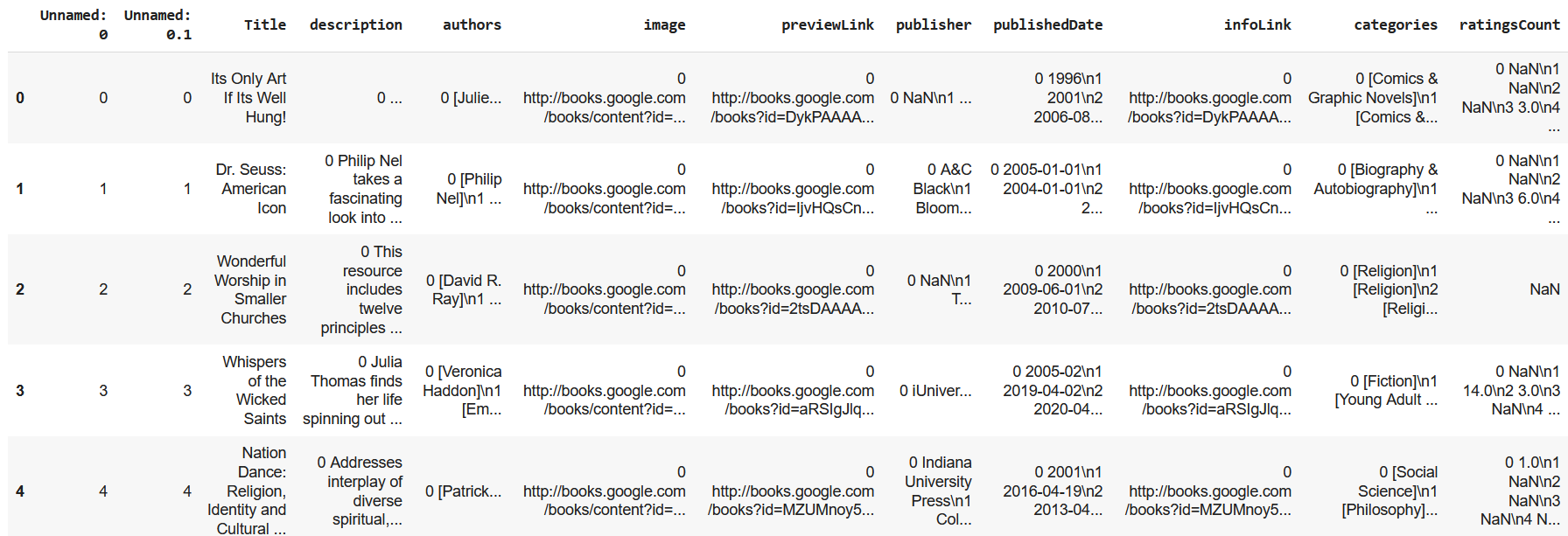


Figure 9 The data set before preparation

* 1. Removing extra columns. two columns out of twelve columns are removed.

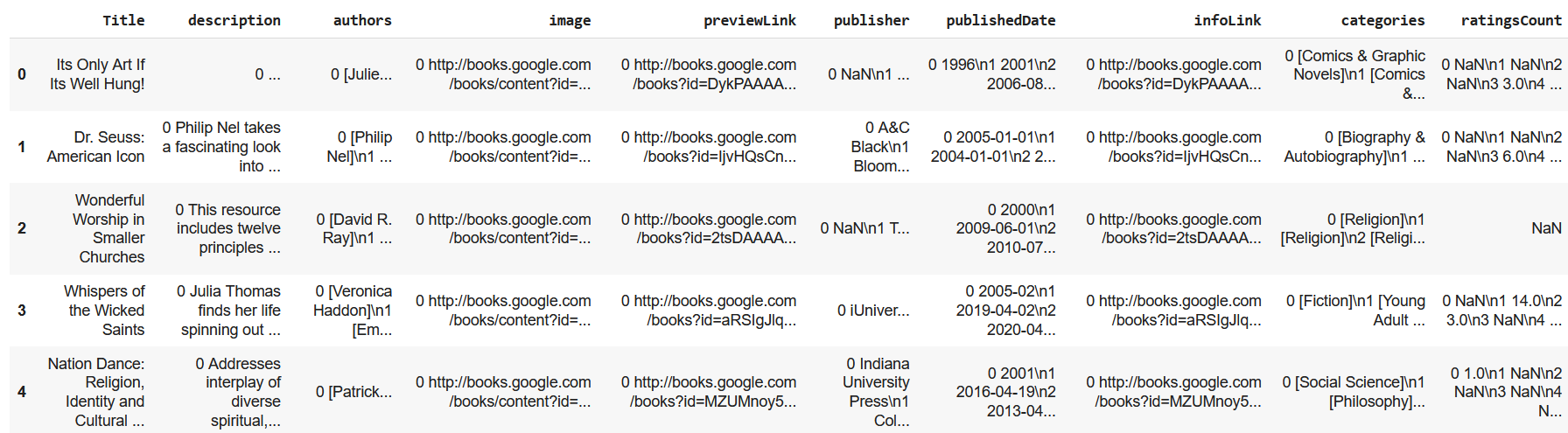


Figure 10 The dataset after removing extra columns

* 1. Removing nulls.
  2. One thousand books are chosen instead of ten thousand to reduce running time.
  3. Cleaning the description text by removing the digits, extra spaces, and special characters.
  4. The description column only will be used to cluster the books.

## Feature Engineering:

We used TF-IDF feature engineering technique to convert descriptions text into numbers.

## Models:

We used K-means model. WCSS and Silhouette will be used to determine which k would be better.

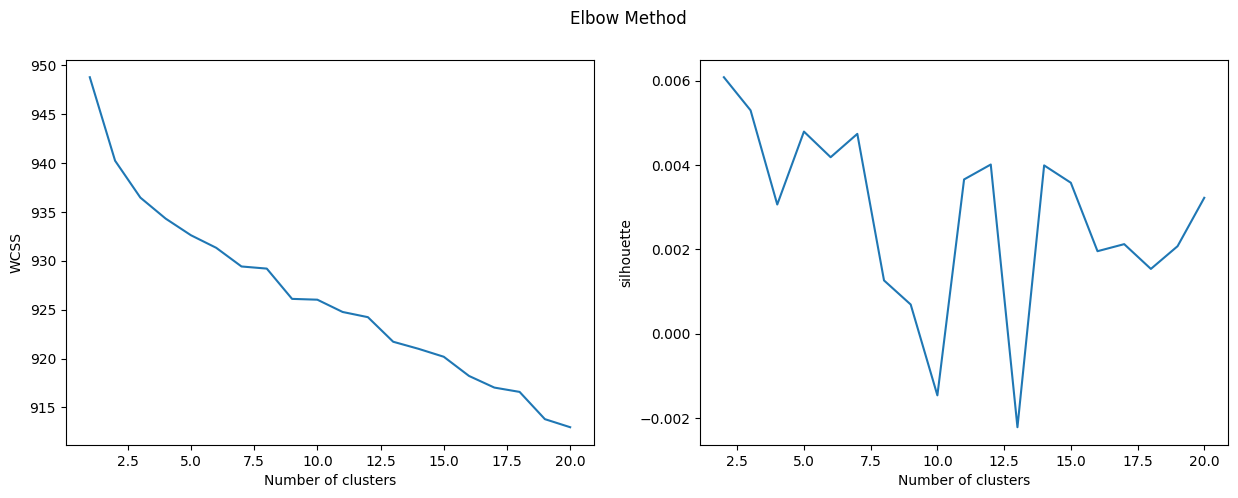


Figure 11 Elbow method with K-means

From the above figure, the chosen **k will be 7**

# Recommender:

In this part, book reviews data from Amazon is used. We have implemented five different recommendation methods:

1. Popularity Based (Top In the whole collection).
2. Recommendation using Average Weighted Rating.
3. User-Item Collaborative Filtering Recommendation.
4. Recommendation using surprise library (specific to a user).
5. Recommendation using surprise library (not specific to a user).

But we used only the last one to connect it with our chatbot. The steps of recommendations are as follows:

## Data Preparation:

* 1. One million reviews are taken from the dataset.
  2. Removing extra columns. Seven columns out of eleven columns are removed.
  3. Renaming columns.
  4. Removing the users that don’t have userID.
  5. Removing duplicated rows.
  6. Constructing the “class” column that have the class of the review (“Positive”, “Neutral”, “Negative”).
  7. Choosing balanced sample from the data (all three classes will have the same number of reviews as the minority class).
  8. Choosing the most frequent 15 books.
  9. Cleaning the reviews text by removing the digits, extra spaces, and special characters.

## Building the recommender:

### Popularity Based (Top In the whole collection):

This is an easy method that relies on counting the number of ratings for each book and recommending the top 15 books with the highest ratings.

Chart

Description automatically generated

Figure 12 Number of ratings for each book

**Recommendation Results are as follows.**

A screenshot of a computer

Description automatically generated with medium confidence

Figure 13 The recommended books

### Recommendation using Average Weighted Rating:

In this method, for all the books we calculated a weighted score using the below formula:

where:

represents the total number of ratings received by the book,

represents the minimum number of total ratings considered to be included,

represents the average rating of the book and,

represents the mean rating of all the books.

using Average Weighted Rating and recommended the books with the highest score.

Graphical user interface, text

Description automatically generated

Figure 14 Recommended books using Average Weighted Rating

### User-Item Collaborative Filtering Recommendation:

Using this method, we could recommend by considering user ratings and finding cosine similarities in ratings by several users.

**Example 1:**

Text

Description automatically generated

Figure 15 The recommended books given "ERAGON: INHERITANCE, BOOK ONE." book as an input

**Example 2:**

**Text

Description automatically generated**

Figure 16 The recommended books given "The Devil Wears Prada" book as an input

### Recommendation using surprise library (specific to a user):

We used the surprise library twice in our code, the first time to implement a recommender regarding a specific user.

**Example:**

If we have a user ID



Figure 17 Example for user ID

The recommendation:

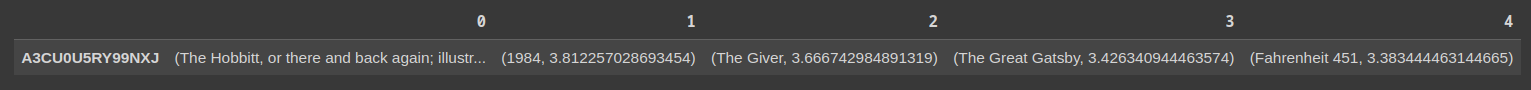


Figure 18 The five recommended books for this specific user

### Recommendation using surprise library (not specific to a user):

In this method, we also used the surprise library to implement a general recommender not specific to a user.

In the beginning, we performed some visualization for the average ratings.

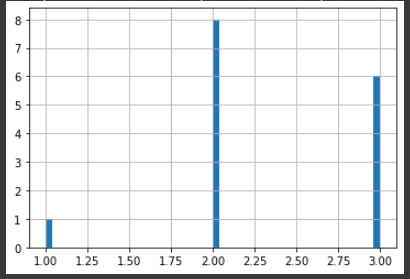


Figure 19 The number of books vs the average rating

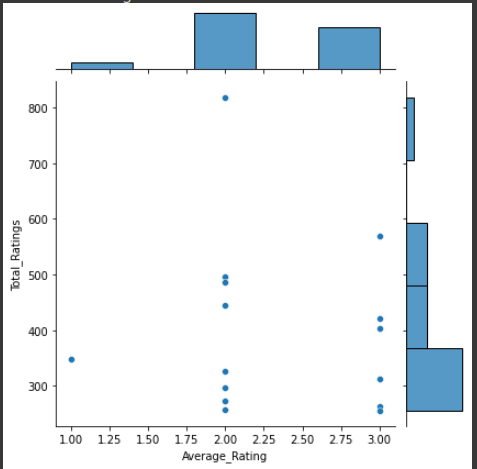


Figure 20 Scatterplot shows the total number of ratings and the average rating for each book

We also used this method to be connected with our chatbot so, we have implemented a function that takes an initial book name and user review classification results for this initial book and returns back the recommendation.

A picture containing text, orange, dark

Description automatically generated

Figure 21 Passing the book name and "Positive" to get the book recommendation



Figure 22 The recommended book

A picture containing text, orange, dark

Description automatically generated

Figure 23 Passing the book name and "Neutral" to get the book recommendation



Figure 24 The recommended book

If you noticed here that we used the same book twice:

The first time with positive feedback from the user

The second time with neutral feedback.

In the two cases, we got different recommendations because while implementing this function we set the return to be the most correlated book if the feedback was positive, and on the other hand we return the least correlated book if the feedback was negative or even neutral.

# Chatbot design:

## Intents design:

Four Intents are developed (“GetName”, “recommender”, “recommender followup”, and “thanking”) besides the two default intents as shown in Figure 25. The Contexts are used to force a specific sequence for the chat. The Webhook call is enabled for “recommender” and “recommender followup” intents. “recommender” intent is responsible for recommending a random book and “recommender followup” is responsible for recommending a book based on the feedback of the user about the previous recommended book.

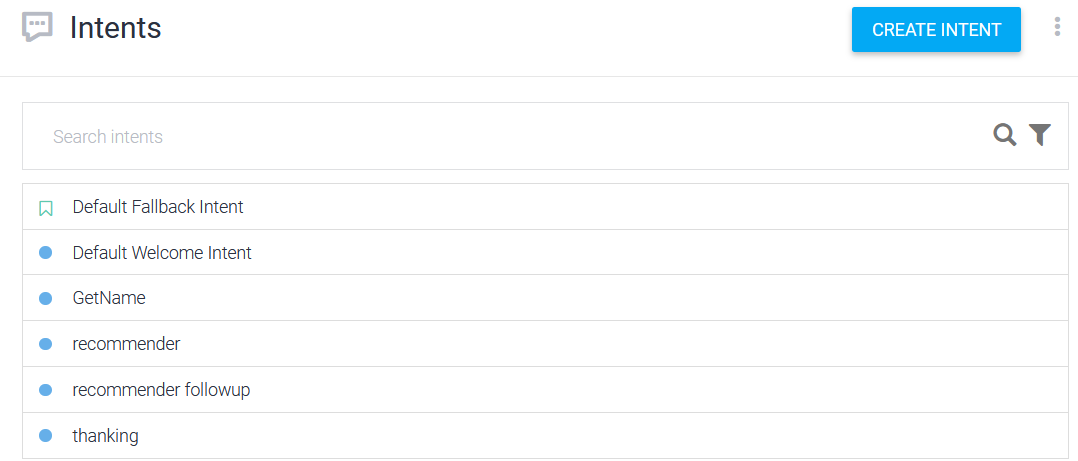


Figure 25 The developed intents

The training phrases for the developed intents (“GetName”, “recommender”, “recommender followup”, and “thanking”) are shown in the following table.

|  |  |
| --- | --- |
| **GetName Intent** | **thanking Intent** |
| The given name by the user will be stored in name parameter. |  |
| **recommender Intent** | **recommender followup Intent** |
|  |  |

The input and output contexts of the intents are shown in the following table.

|  |  |
| --- | --- |
| **Default Welcome Intent** | **GetName Intent** |
|  |  |
| **recommender Intent** | **recommender followup Intent** |
|  |  |

The responses of the intents are shown in the following table.

|  |  |
| --- | --- |
| **Default Welcome Intent** | **GetName Intent** |
|  | The chatbot will use the name parameter to call the user by his name. |
| **recommender Intent** | **recommender followup Intent** |
| This response will be generated only if there is a problem in the Webhook call. | This response will be generated only if there is a problem in the Webhook call. |
| **thanking Intent** |
|  |

The URL provided by NGROK will be put in the fulfilment as shown in the following figure.

Graphical user interface, text, application

Description automatically generated

Figure 26 The fulfillment

## Evaluation (overall system):

|  |  |
| --- | --- |
| The triggered intent, output context, and the chatbot response are as intended. | The triggered intent, output context, and the chatbot response are as intended. |
| The triggered intent and output context are as intended. The chatbot managed to recommend a random book for the user. One drawback of the chatbot, that it does not ask the user about his feedback about the recommended book. | The triggered intent is as intended. The chatbot managed to recommend a book for the user based on the feedback of the user about the previous book. |
| The triggered intent and the chatbot response are as intended. |

Text

Description automatically generated

Figure 27 The book name and the user feedback that the “recommender followup” intent took and the classifier prediction of the feedback

From the above figure, it is concluded that the classifier model managed to classify the feedback from the user correctly.

## Error Analysis (overall system):

|  |  |
| --- | --- |
|  |  |

In the previous example, the classifier did not manage to predict the class of the feedback correctly (predicted “Neutral” for obvious “Negative” class). So, the classifier needs to be improved as its accuracy is not high enough.

Graphical user interface, application

Description automatically generated

Figure 28 The response of the chatbot for the user feedback

From the previous figure, it is concluded that the “recommender followup” intent had not triggered, and the Default Fallback Intent is triggered instead. So, the training phrases of the “recommender followup” intent need to be improved.

# References:

1. J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. RecSys, 2013.