Recommender System Machine Learning Project for Beginners-3
Project Overview

Business Overview

Imagine you want to buy a gift for your dear ones. You opened an eCommerce website on your phone and started searching for the gift. After a few minutes, you find a section which shows very similar gifts you wanted. Now you must be thinking, how do they know similar items based on one thing? Most platforms use the Recommendation system at the backend to show you the best items based on your search history. There are several choices in the digital market world, which are somehow too overwhelming for the user to choose from. There is always a need to filter and prioritize the relevant items for each user to engage the customers with the platform, which eventually impacts the revenue. Content-based filtering is one of the most used techniques to personalize the content for each user. It uses previous actions and feedback about users' liking to provide them with similar recommendations.

As a part of a series of Recommender system projects, this project covers Recommendations using a wide variety of Content-Based Filtering algorithms in Python. It also demonstrates one of the developed algorithms using the Streamlit app framework. If you haven't already visited, here is the previous project of the series [Recommender System Machine Learning Project for Beginners-2]

Aim

To build a Recommender System using various content-based filtering techniques and similarity measures and to create a web application for the same using Streamlit

Data Description

The dataset contains about 4k products in various categories for a UK-based non-store online retail business. The company mainly sells unique all-occasion gifts with maximum wholesaler customers.

Tech Stack

- ➔ Language: Python
- ➔ Libraries: pandas, NumPy, seaborn, matplotlib, re, gensim

Approach

1. Data Description
2. Data Preprocessing
3. Ranking Functions
4. Building Recommendation system
    a. Count Vectorizer
    b. TFIDF Vectorizer
    c. Word2Vec
    d. FastText
    e. Glove
    f. Co-occurrence Matrix
5. Streamlit Web App

Modular code overview:

```
input
|_data.xlsx

lib
|_Content based recommendations.ipynb

output
|_recommendations.xlsx

src
|_engine.py
|_product_recommendation_streamlit.py
|_config.ini
|_ML_Pipeline
            |_distance.py
            |_load_models.py
            |_preprocessing.py
            |_recommendations.py
            |_utils.py

readme.md
requirements.txt
```

Once you unzip the modular_code.zip file, you can find the following folders within it.

1. input
2. src

3. output
4. lib
5. requirements.txt

1. The input folder contains the data that we have for analysis. In our case, it contains data.xlsx

2. The src folder is the heart of the project. This folder contains all the modularized code for all the above steps in a modularized manner. It further includes the following.
    a. ML_pipeline
    b. engine.py
    c. config.ini
    d. product_recommendation_streamlit.py

The ML_pipeline is a folder that contains all the functions put into different python files, which are appropriately named. These python functions are then called inside the Engine.py file.

The config.ini file is used to store the variables which are called in engine.py file. The product_recommendation_streamlit.py file has the code for streamlit UI.

3. The output folder contains the excel sheet containing the top 10 recommendations for the selected product with the score

4. The lib folder is a reference folder, and it contains the original ipython notebook that we saw in the videos

5. The requirements.txt file has all the required libraries with respective versions. Kindly install the file by using the command pip **install** -r **requirements**.**txt**

6. **All the instructions for running the code are present in readme.md file**

Project Takeaways

1. What is a Content-Based Recommender System?
2. What are the components of the Content-Based Recommender System?
3. How to convert Text to Features?
4. What are Similarity measures?
5. Different types of Similarity Measures

6. Understanding Manhattan, Euclidean, Cosine Similarity
7. How to Rank the Recommendations?
8. Recommendations using Count Vectorizer
9. Recommendations using TFIDF Vectorizer
10. Recommendations using Word2Vec
11. Recommendations using FastText
12. Recommendations using Glove
13. Recommendations using Co-ocuurence Matrix
14. Advantages and Disadvantages of Content Based filtering
15. Building web application using Streamlit