**Recommender System for Airbnb**

Personalisation and Machine Learning

Jerome Goldstein

Student Id: 21036100

Github repo: <https://github.com/jeromegold/personlisation>

Word Count: 1007 words

Index

Introduction

Aim

Recommender System Overview

Recommender System Concepts

Implementation of an Airbnb Recommender

Conclusion

References

**Introduction**

As the internet has developed and large amounts of data can be transferred at high speed, we can listen to music, watch movies and browse for fashion in real time. With the increased accessibility of the internet, it was quickly recognised that the choices and preferences of internet users could be recorded.

This stored information comprising user details and their preferences allows for bidirectional information transfer. Companies such as Meta, Google and other smaller businesses can collect data on users’ profiles that can better inform product development. Additionally, users can be provided with choices that match their prior recorded preferences.

Machine learning techniques have been developed to provide the computational basis for recommender systems.

**Aim**

In this project the aim is to experiment with the techniques involved in developing a recommender system, specifically content based filtering and collaborative filtering, for the rental site Airbnb.



**Recommender System Overview**

The idea behind recommender systems is relatively simple. Data is collected on user preference and this data is then used as the basis for providing consumers and users of the internet with choices that align with their prior selections.

Recommender systems not only provide paying consumers with choices that closely match their preferences, but these systems can also be implemented to help sift through thousands and millions of potential options in spheres beyond paid consumerism. For example, there are billions of videos on Youtube and when opening Youtube’s homepage there are videos tiled across the screen.

Youtube could randomly select any set of videos for any particular user. However, Youtube uses a recommender system to select specific videos to populate the home page based on viewing history.

This recommender system was outlined in 2016 in an article titled, Deep Neural Networks for YouTube Recommendations, by Covington, et al.

The major branches in recommender system implementation are content based filtering, collaborative filtering and deep neural network (Figure 1).

Graphical user interface, application, Word

Description automatically generated

**Figure 1.** Recommender System types according to Google

<https://developers.google.com/machine-learning/recommendation/overview>

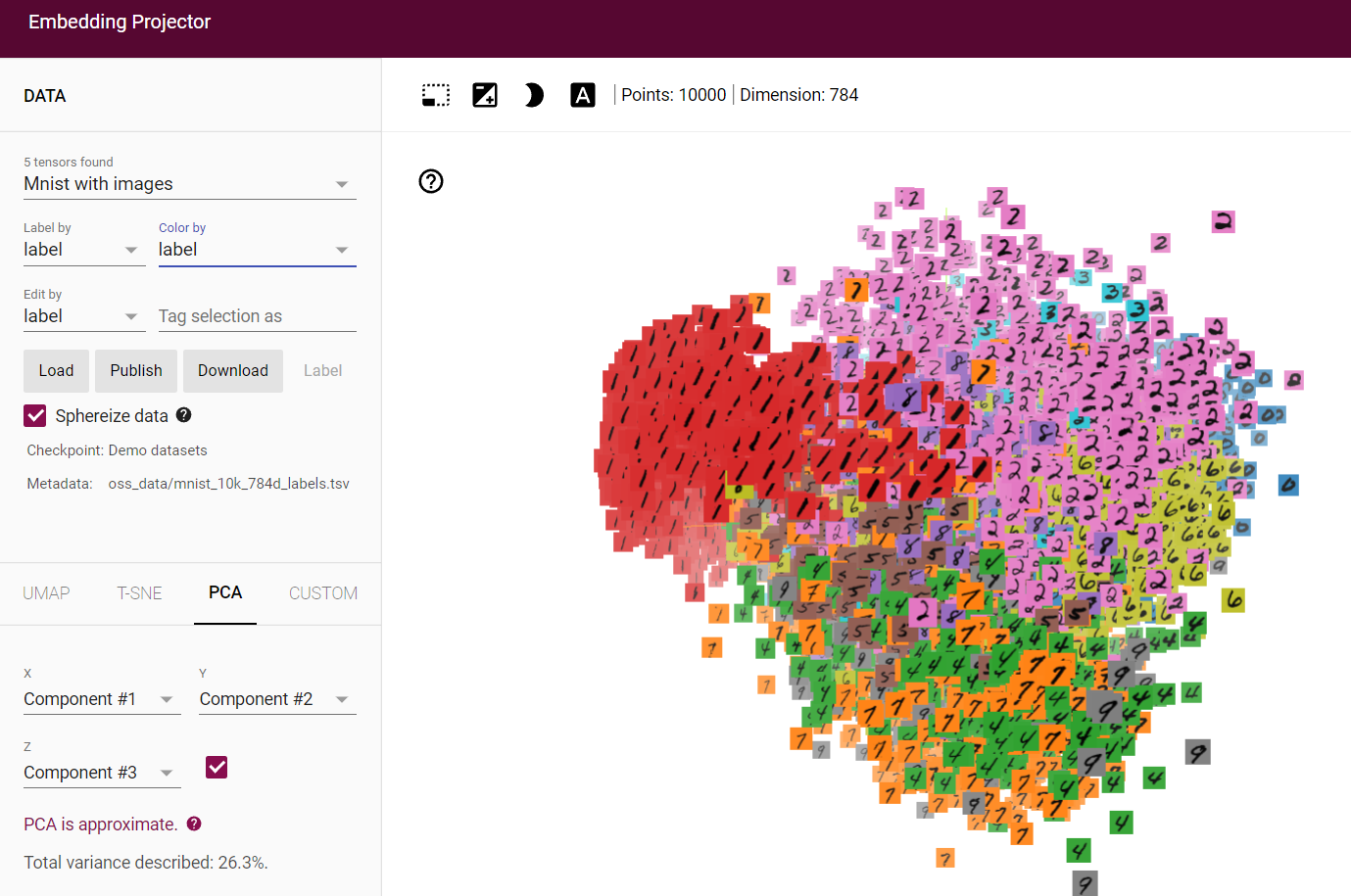
**Recommender System Concepts**

One of the first steps in recommending is filtering relevant options from a larger dataset. The two most common filtering options are content based filtering and collaborative filtering.

Content based filtering refers to the recommendation of a similar item based on the user's preference for that item in the past. Collaborative filtering, as the name implies, is a method of recommending items based on similarities between users/collaborators. The idea is that if two individuals have similar features such as age, gender or location then recommendations can be made based on similarity in those features.

The basis for recommenders and filtering is the concept of an embedding space. Mathematically, the embedding space is represented by E -> Rd. The general concept of a mapping is that a set can be mapped to a second set where the information housed within the second set can be used to reach judgments and infer other patterns.

Figure 2 represents an embedding space where the MNIST dataset is mapped as vectors and each integer is represented by a colour demonstrating that that location of the integers in a “space” can be used to try and identify when a similar image is close by.



**Figure 2.** Visual representation of embedding space of the MNIST database

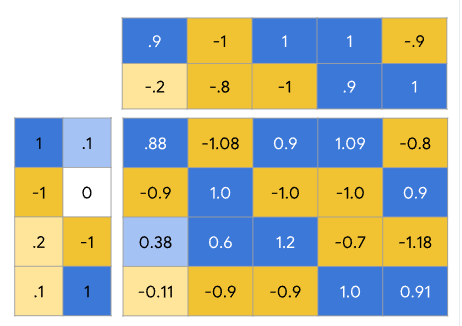
with images as embedding vectors using Tensorflow’s Embedding Projector

https://projector.tensorflow.org/

There are several different ways to measure how close items are to each in other in an embedding space. These include using the cosine angle between two of the embedding vectors, calculating the dot product of two embeddings or determining the Euclidean distance between two embeddings.

The general idea is to determine some form of metric or “distance” between two vectors and thereby quantify similarity. When the embedding vectors are normalised the dot product, cosine and Euclidean distance have the same value.

Collaborative filtering is also associated with the concept of matrix factorisation. The matrix below is an example of a matrix factorisation where rows represent users and columns represent items.



**Figure 3** A matrix factorisation that represents the combination of a user matrix and item matrix.

https://developers.google.com/machine-learning/recommendation/collaborative/matrix

There is an additional alternative to content and collaborative embeddings. Recommendations can also be arrived at using a deep neural network and models such as softmax. This was not implemented in this project.

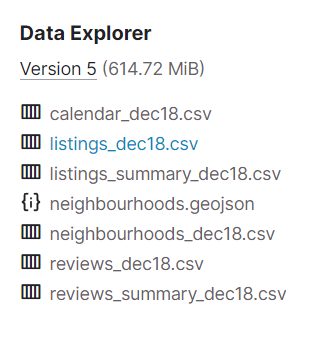
**Implementation of an Airbnb Recommender**

Developing a recommender for Airbnb rentals requires a number of steps. The first was to find an appropriate dataset and to determine what feature and measure would be used to conduct the recommendation.

There are a number of datasets housing Airbnb information on Kaggle ([www.kaggle.com](http://www.kaggle.com)). Some of the possible recommender options included recommending rentals based on Airbnb ratings or filtering out the various rental options based on rental description.

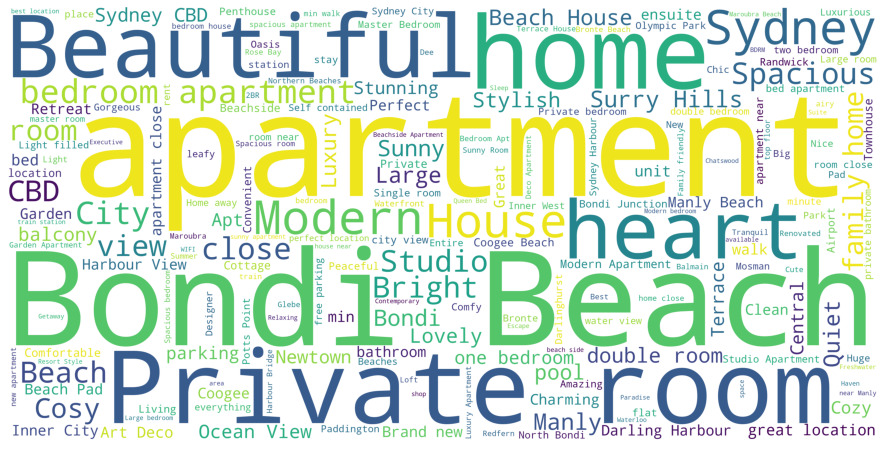
Irrespective of the specific nature of the recommender system, the principles remain the same. For content filtering, a set of embedding vectors needed to be established along with an embedding space. From there a method needed to be chosen (eg. cosine, dot product, etc.) to determine the “distance” between vectors and then a mechanism needs to be defined to return the recommendations to the user. And for collaborative filtering a matrix factorisation model needs to be created.

The dataset used for the recommender was the Sydney Airbnb Open Data (<https://www.kaggle.com/datasets/tylerx/sydney-airbnb-open-data?select=listings_dec18.csv>).



The jupyter notebook available on Kaggle <https://www.kaggle.com/code/rdaldian/airbnb-content-based-recommendation-system> was referred to in the implementation of this recommender.

The first step was to select the required columns out of the 96 columns provided in the dataset. Using the python package “Wordcloud”



**Discussion**

**Conclusion**

In summary, in this project I investigated various techniques for the implementation of the

**References:**

1. Seeda, P. A Complete Guide To Recommender Systems — Tutorial with Sklearn, Surprise, Keras, Recommenders. Oct 2021. https://towardsdatascience.com/a-complete-guide-to-recommender-system-tutorial-with-sklearn-surprise-keras-recommender-5e52e8ceace1
2. Google Machine Learning Crash Course. https://developers.google.com/machine-learning/recommendation
3. Paul Covington, Jay Adams, Emre Sargin. Deep Neural Networks for YouTube Recommendations. Sep 2016. Google Mountain View, CA