COMP4332 / RMBI 4310 Project 3 Report (Rating Prediction)

Team Members: SETHI Aryan (20634962), CHOW Hau Cheung Jasper (20589533), KAUSHAL Kaustubh (20634039), PAREKH Yashasvi Gopalbahi (20634089)

Team Name: AKJY **Group ID:** 05

Introduction

The rise in popularity and adoption of the internet and the sheer amount of data it has brought has allowed businesses to get an insight into consumer behaviour like never before. From analysing the retention rate on a YouTube video to determining the click-through-rate for an e-Commerce website, leveraging the data provided by the Internet is vital for a company's success nowadays. This is especially important in current times due to the countless number of options available for users to enjoy whatever they want to, and businesses need to ensure that they are at the forefront of being the top recommendation for users.

Using a recommendation system allows companies such as eCommerce platforms like Amazon and streaming services like Netflix to understand the behaviour of a consumer on their platform and suggest similar products they may enjoy to maximise the users' time spent on the platform. Recommendation systems are in fact one of the key tools that allowed Netflix to become a market leader in the streaming industry. In this project, we aim to predict users' ratings on certain items given a sparse user-item rating matrix. The overall performance of the model built will be measured using the Root Mean Square Error (RMSE) between the predicted ratings and the actual rating values in the test set.

Preprocessing and Exploratory Data Analysis

The provided data includes user ratings, user information and business information. In particular, information about 2980 users and information about 5964 businesses is provided. User information includes: *'user_id', 'name', 'review_count', 'yelping_since';* reactions like *'useful', 'funny'* and *'cool'; 'elite', 'fans', 'average_stars'* and several categorical columns for different types of *'compliments.'* Business information includes: *'business_id', 'name', 'address', 'city', 'state', 'postal_code', 'latitude', 'longitude', 'stars', 'review_count' 'is_open', 'attributes', 'categories'* and *'hours'*. For the user ratings, 60080 ratings are provided in the training dataset, 7510 ratings are provided in the validation set, and 7510 ratings are provided in the test set.

To construct a robust recommender system, we require the model to be able to take in this large dataset and memorise different users' frequently occurring behaviours on the platform. However, since the ratings matrix is sparse, memorisation cannot generalise well to unrated user-item pairs. Therefore, the model must project the high-dimensional sparse feature vectors into a lower-dimensional space to get feature embeddings, which can make more generalised predictions. This is precisely what the 'Wide & Deep Learning Model' (WDLM) allows us to do, as it jointly trains a wide linear model for memorisation and a deep, feed-forward neural network for generalisation to combine the benefits of the two approaches for real-world recommendation systems.

In order to use the model, there is some data preprocessing required for feature extraction. The deep, feed-forward component of WDLM takes as input: i) the continuous-valued features (such as *"review_count"*), and ii) the lower-dimensional embeddings of the categorical features. (ii) can be obtained by extracting the category for each feature, and then mapping each value of that feature's category to a distinct embedding vector. The wide component of WDLM then takes as input the cross product transformations of the most frequently co-occurring categorical features.

Feature Engineering & Data Modelling

Firstly, the data from user.csv and business.csv were imported into dataframes as they originated from different csv files. The columns were also renamed by adding the prefixes **user_** and **item_** respectively. The data was then separated into continuous features and categorical features. The continuous features are listed below:

- user_average_stars
- user_cool
- user_fans
- user_review_count
- user_useful
- user_funny
- item_is_open
- item_latitude
- item_longitude
- item_review_count
- item_stars
- user_compliment_hot
- user_compliment_more
- user_compliment_profile
- user_compliment_cute
- user_compliment_list
- user_compliment_note
- user_compliment_plain
- user_compliment_cool
- user_compliment_funny
- user_compliment_writer
- user_compliment_photo

The different scales in the data were addressed by performing scaling. Standard scaling was used on each continuous-valued feature by removing the mean of the training samples and scaling to unit variance.

For the categorical data, "*item_attributes"* was particularly useful as it had extra features about the related business. The categorical features were extracted from this column, and the total set of categorical features used is listed below.

- item_city
- item_postal_code
- item_state
- business_id
- item_RestaurantsAttire
- item_WiFi
- item_Ambience
- item_NoiseLevel
- item_BusinessAcceptsCredit
- item_RestaurantsPriceRange
- item_GoodForMeal
- item_BusinessParking
- item_RestaurantsGoodForGroups
- item_Alcohol
- item_RestaurantsReservations
- user_id
- user_elite

COMP4332 / RMBI 4310 Project 3 Report (Rating Prediction)

Team Members: SETHI Aryan (20634962), CHOW Hau Cheung Jasper (20589533), KAUSHAL Kaustubh (20634039), PAREKH Yashasvi Gopalbahi (20634089)

Team Name: AKJY **Group ID:** 05

The set of 15 item-based categorical features and 2 user-based categorical features were used to generate both the deep categorical features (via one hot encoding of each feature) and the wide features. The set of wide features were prepared by reviewing the most frequent combinations of k categorical features.

For k=1, choose top 500 combinations, For k=2, choose top 50 k-category combinations

For k=3, choose top 30 k-category combinations, For k=4, choose top 20 k-category combinations

Lastly, all the continuous features, deep categorical features and wide features were concatenated into an input of size (60080, 601) where each training example is a 601-dimensional vector.

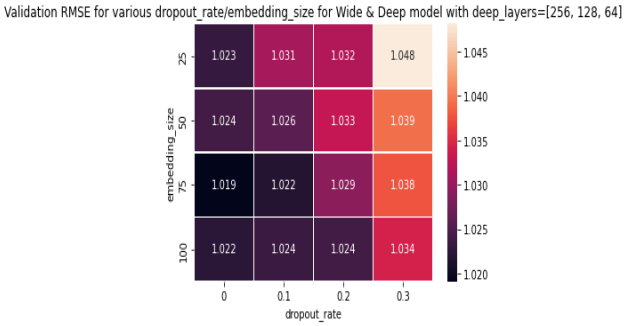
Hyperparameter tuning:

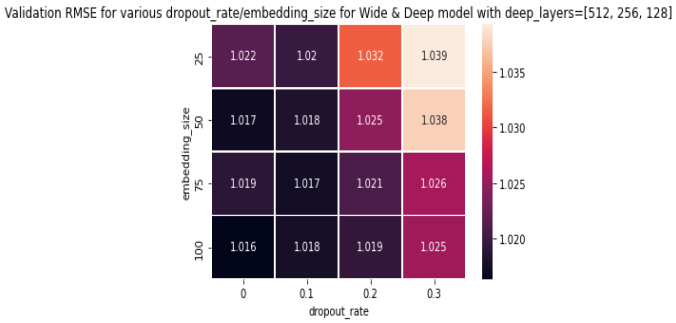
We used the Adagrad optimizer and trained for 5 epochs to test each of the 4 × 4 × 3 following combinations of the following hyperparameters:

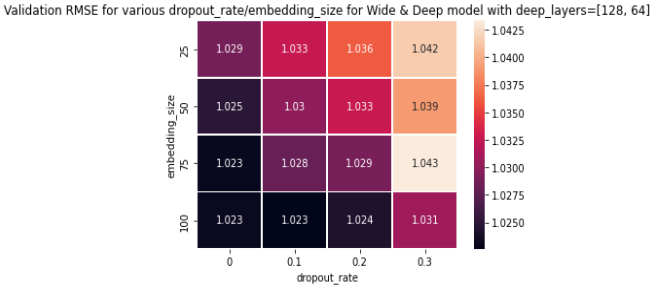
Embedding sizes of each deep categorical feature: **25, 50, 75, 100**

Dropout rates: **0, 0.1, 0.2, 0.3**

Deep component layers*: **(256, 128, 64), (512, 256, 128), (128, 64)**







*In general the MLP in the deep component of the WDLM is given in the form $(c_1, c_2, ..., c_t)$ where c_i describes the output size of the ith layer and t is the total number of MLP layers.

We analysed the validation RMSE values to pick the best hyperparameters. Based on this, we found that the lowest validation RMSE was achieved with embedding size = 100, dropout rate = 0, deep component layers = (512, 256, 128). Additionally, we created heatmaps of validation RMSE to further understand interesting trends for each hyperparameter (assuming other hyperparameter values remain the same).

In each of the three heatmaps below, the lowest RMSE for each was observed when dropout rate = 0, and RMSE tends to increase as dropout rate increases. Furthermore, it can be observed that generally the higher the embedding size, the lower the validation RMSE. However, in the first heatmap, the lowest RMSE was when embedding size = 75, but the lowest RMSE was when embedding size = 100 in both the second and third heatmaps. Therefore, there are diminishing returns, i.e. increasing the embedding size further is unlikely to improve model performance. In terms of the structure of the MLP, by comparing the third heatmap (2 layers) with the 1st and 2nd heatmaps (3 layers) we noticed that the deeper the model, the lower the RMSE is. Finally, by comparing the 1st and 2nd heatmaps, which both have 3 layers but different output sizes in each layer, we observe that the validation RMSE is lower when the output sizes in each layer of the MLP is larger..

Conclusion

In conclusion, we were able to successfully train a wide and deep model to predict the ratings of users for businesses by constructing a variety of categorical and continuous features. Additionally, by tuning hyperparameters such as the dense layer sizes in the deep model, dropout rate and the embedding size, we achieved a validation RMSE of 1.014, a significant improvement over the baseline of 1.09. Further work to improve performance may involve tuning the number of wide features to use (i.e. adjusting the number of most frequent k-category combinations for different k,) or using residual connections in the deep component of the model to leverage the increased expressive power of nested function classes.