Report

Created by: Himanshu Aggarwal

AIM

To generate personalised relevant venue ranked lists corresponding to user sessions, to increase conversion rate.
Basically, we want a ranked list of venues for each user session, based on the earlier engagements of users in the sessions.

Technical Details

All development is done on Python 3.8 on MacOS.
Please refer 'requirements.txt' for package details.

Implementation is in the form of **jupyter notebooks**, since representation is easier that way. However, I have stored HTML versions of the notebooks for easy viewing :

- For **EDA notebook,** please go to **'/reports/eda.html'** or **'/notebooks/eda.ipynb'**
- Please find the **implementation code** as a **notebook** at '**/notebooks/ranking_model.ipynb**', or as **HTML doc** at '**/reports/ranking_model.html**' for easy viewing.

Exploratory Data Analysis

Main observations are as below:

- Session Data: Contains 1369807 engagements of users in 4415 sessions
- Venue Data: Contains 1215 venue and their associated meta-data
- 'sessions' dataset is imbalanced data with much more negative (purchased=False) than positive samples (purchased=True)
- Both datasets have 'NaN' values present. These attributes are too important to lose, hence require some form of value imputation
- There is mostly only single conversion (purchased=True) per session
- All positive engagements (purchased=True) in 'session' data, have either of the following attributes as True - 'has_seen_venue_in_the_session', 'is_from_order_again', 'is_recommended'
- 'Session' data contains 19149 duplicate rows which are mostly negative samples. These can be dropped, since negative class is already in majority and keeping them will not benefit in model creation.
- There doesn't exist any major correlation between the attributes in both the datasets

Assumptions

- There is no ground truth for ranks, it is purely derived by past purchases of the user and global user behavior
- All 1215 venues are accessible to all users in all sessions
- Ranked results should contain all venues
- In real time scenario, user's data (which includes 'is_new_user', 'is_from_order_again', 'is_recommended') is accessible to the ranking algorithm, i.e. historical data of which venues the user has already ordered from, and have been recommended
- Relevance of the venue for a user session is defined by purchase behavior, i.e. 'purchase = True' is considered a positive sample

Evaluation Metric

- **Mean Reciprocal Rank (MRR)** is used to compare the performance of the models and the quality of the ranked lists
- Since there is mostly a single positive sample per session, so evaluating the results based on how well that positive sample is ranked, is only logical
- Other evaluation metrics used for ranking evaluation are NDCG and MAP, but they are not useful when we know relevance for only a single item

Modeling a Ranking Algorithm

Baseline Model

Two types of non-ML baselines have been implemented - sorting-based and nonlinear:

- Sorting-based model consists of sorting the venues based on 'popularity' and 'rating'
- Nonlinear model consists of a weighted average of the nonlinear outputs of 'popularity', 'rating' and 'price_range'
- Consequently, all user sessions have the same ranked list of venues based on global behavior and popularity

ML Models

Pipeline contains following sections:

- Data cleaning and Preprocessing:
    - removing duplicate data
    - imputing missing values
    - upsampling for the minority class do manage bias (optional)
    - scaling the real-valued attributes, so that each feature contributes proportionately equally in the model training

- Feature Selection:
    - since there is not visible correlation between attributes, automated selection is not possible
    - Manual selection of features is done from 'session' data considering what data can be available in real time (mentioned in assumptions)

    - ‘position_in_list’ and ‘has_seen_venue_in_this_session’ are not used while modeling since such information is only available once user in on the platform and not beforehand

- Modeling:
    - Ranking problem is modelled as a classification problem, where the venue features and few selected session features are treated as training features and ‘purchased’ attribute is the target
    - All engagement rows are used in this stage irrespective of the session
    - Train-test splitting of data is done as 70:30 split
    - Multiple scikit-learn algorithms have been trained and compared - Logistic Regression, Decision Trees, Neural Nets, Random Forest
    - This classification task is evaluated with precision and recall, since accuracy is not a good measure with imbalanced dataset

- Ranking Venues
    - The classification probabilities from the trained models are calculated for each venue and each user session separately
    - All the venues are ranked separately for each user session

- Ranking Evaluation
    - The ranked lists for each user session is evaluated based on the original purchase in that session, i.e. relevant sample item in the ranked result is the one that was originally purchased
    - Intuition behind this is that the model learns the global purchase behavior from the data and then rank the venues based on past engagements with the venues from a particular session
    - MRR is calculated for each ranked list
    - Baseline and ML-models, both are evaluated the same way

Results

Following are the results from training **classification** models (test data):

MODEL	PRECISION	RECALL	ACCURACY
Logistic Regression	0.5945	0.1594	0.996
Decision Tree	0.6081	0.1983	0.9968
Random Forest	0.5966	0.2108	0.9968
Neural Nets (Shallow)	0.6385	0.1388	0.9968

Following are the **ranking evaluation** results from all the models:

TYPE	MODEL	MEAN RECIPROCAL RANK (MRR)
Baseline	Sorting-based	0.00927935757757337
	Nonlinear Model	0.011055931991708578
ML Models	Logistic Regression	0.480392248361908
	Decision Tree	0.5367197714992566
	Random Forest	0.5423520574954974
	Neural Nets (Shallow)	0.48771859319948674

Discussion and Future Scope

Observations and decisions

- Ranking model is based on **pointwise Learning-to-Rank** approach
- Data has imbalance classes can introduce **bias** and lead to model overfitting
    - Much more negative samples than positive
    - It can be handled as following:
        - Choosing better evaluation metric (precision, recall and not accuracy)
        - Upsampling the minority class (not downsampling as we don't want to lose information from the negative samples)

- **Upsampling** did not affect the ranking results very much

What else can be done?

- Using cross-validation, with straightified split can train models better
- Hyper-parameter tuning using grid search
- Using advanced LTR methods like **pairwise** (RankNet etc.) and **listwise** (ListNet etc.) approaches
- If we have more sessions corresponding to a user, we can personalize the ranked lists more, by understanding user context (location, TOD, DOW), preferences in terms of cuisine, ratings, delivery time, etc.

    - User profiles can be created and used to represent user preferences
- For actual real time ranking of venues, user behavior can be observed on initial few rankings (pagination) and then further list can be re-ranked accordingly.
    - We can model this as Markov Decision Process - states being user context, actions being new recommendations, rewards being +1 if user engages, -1 if not and +100 for conversion
- Features that can be added:
    - user context attributes (location, day, time)
    - average billing amount
- Evaluation: Metrics like NDCG and MAP can be used if we have multiple sessions (multiple conversions) corresponding to a user