Proposal for Machine Learning Engineer capstone project: Starbucks Challenge

Domain background

Historically, marketing has always been catered to groups of people through mass-media; make-up and fashion through ads in magazines targeted to young women, beer and car ads on tv during sports events, massive light-boards on Times Square bombarding anyone and everyone with promises of better lives, the list goes on. These ads are seen as intrusive and obnoxious and are quickly filtered out as noise or even actively avoided.

> The reason we're blocking ads is that in the context those ads are being delivered, irrelevant ads are – quite literally – painful. Even relevant ads have a very high threshold to get over. [1]

In the digital age though, there is a unique opportunity to fully personalize potential customers' experiences; we know how old someone is, a rough estimation of their income, spending behavior and the ads they see are on a medium that enables a for-your-eyes-only experience. This enables companies to not only better target campaigns, but also actually offer increased value to their customers by presenting them with offers where they want them, when they need them.

This domain is personally relevant because in my day-job I work on experience personalization, though only on the output and analytics gathering side. With this project I want to get more into the back-end of things, optimizing and personalizing propositions into offers that aren't just seen as less obnoxious, but that actually help customers get what they need when they need it. Offering them increased value to their interaction with the company and as a result making them into more loyal customers and even promotors of the company.

Problem statement

Starbucks wants to personalize the delivery of offers for its customers in order to maximize money spent after said offer is presented.

Datasets and inputs

The data is split up into the offer portfolio, user profiles and customer interaction transcripts, taken from the Starbucks Capstone Challenge overview:

- portfolio.json - containing offer ids and meta data about each offer (duration, type, etc.)
- profile.json - demographic data for each customer
- transcript.json - records for transactions, offers received, offers viewed, and offers completed

Here is the schema and explanation of each variable in the files:

profile.json Rewards program users (17000 users x 5 fields)

- gender: (categorical) M, F, O, or null
- age: (numeric) missing value encoded as 118

- id: (string/hash)
- became_member_on: (date) format YYYYMMDD
- income: (numeric)

portfolio.json Offers sent during 30-day test period (10 offers x 6 fields)

- reward: (numeric) money awarded for the amount spent
- channels: (list) web, email, mobile, social
- difficulty: (numeric) money required to be spent to receive reward
- duration: (numeric) time for offer to be open, in days
- offer_type: (string) bogo, discount, informational
- id: (string/hash)

transcript.json Event log (306648 events x 4 fields)

- person: (string/hash)
- event: (string) offer received, offer viewed, transaction, offer completed
- value: (dictionary) different values depending on event type
    - offer id: (string/hash) not associated with any "transaction"
    - amount: (numeric) money spent in "transaction"
    - reward: (numeric) money gained from "offer completed"
- time: (numeric) hours after start of test

In data exploration, roughly 12% of profiles were found to have missing values for gender, age as well as income. This may be missing data, but it may also be a cultural effect; people not willing to fill in their profile to maintain a sense of anonymity. During training these records will be explored as a group to be kept as their own demography or simply to be discarded.

Solution statement

The solution should take profile features as input and predict the best offer and channel based on expected money spent. To this end, the input will be put through a preprocessing step to normalize the input and the model output will be presented as a top-n of offer/channel combinations with the expected money spent in reaction to those offers.

This enables the model to be used as an API endpoint for real-time offer selection when a customer is identified in a channel or when an e-mail batch is being sent out.

Benchmark model

The problem stated is very similar to the so-called collaborative filtering personalization. In stead of product recommendations we're presenting people with offers and in stead of recommendation/review scores we're looking at money spent, but in essence the problems are similar. Collaborative filtering is well-suited to neural network models and they have been used in the past to implement such recommendations [2].

Evaluation metrics

Model success will be measured by a combination of Log Loss, as this metric works well at measuring multi-class classification quality by penalizing erroneous classification based on uncertainty of the predictions, and F-1 score to measure the precision and recall. [3]

Log loss:

$$\frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} \times \log(p_{ij})$$

where:

- y_ij indicates whether sample i belongs to class j or not
- p_ij indicates the probability of sample i belonging to class j

In the context of the problem at hand, it pushes the model towards the right money spent prediction per offer/channel.

F-1:

$$\frac{tp}{tp + \frac{1}{2}(fp + fn)}$$

where:

- tp, fp are the number of true resp. false positives
- tn, fn are the number of true resp. false negatives

Positives and negatives will be measured by the outcome of the neuron, >=0.5 is positive, <0.5 is negative.

Project design

Some data wrangling will have to be done;

- A 'no-offer' offer will be added to help identify demographics that will buy products even without being influenced by offers.
- Transactions will be grouped into a new dataset, containing
    - Time from offer received to offer viewed (if any);
    - Time from offer viewed to offer completed (if any);
    - Total money spent between offer viewed and offer completed;
    - Existing features from the profile and portfolio datasets pertaining to the offers and transactions.
- Offers that were presented but weren't viewed but nonetheless completed will be set to have no resulting spending; even though customers used the offer, they weren't aware of it and thus it may be concluded the offer did not influence their behavior.
- Categorical features (gender, channels, offertype, event) will be one-hot encoded.
- Continuous features (money spent, time to offer viewed/completed, age, account age, income) will be normalized to 0-1 range.
- Feature selection will be performed to identify the most relevant features and which features can be safely omitted to simplify the model and optimize training time.
- Lastly, the data will be split based on k-fold cross-validation to obtain a better evaluation of the model even with the relatively low number of datapoints in the set. [4]

The proposed model to be trained is a neural network using the Keras libraries, taking in profile features and predicting the best offer and channel combination based on expected money spent. This model is chosen to make the best use of the continuous nature of the offer performance metric (money spent) and to be able to fully personalize the customer experience in stead of predicting target demographics for offers.

Finetuning of the model will be done through automated hyperparameter tuning, tuning the amount of hidden layers and number of neurons, learning rate and epochs. The focus should ideally be on creating a simple feed-forward model in stead of a complicated multi-layer perceptron in order to optimize training and prediction time.

Discussion

Although this is not in scope for this project, ideally the model should be monitored after deployment:

- Model performance and automated updating of the model based on new data should be implemented;
- The technical performance needs to be monitored to be able to serve the many users of the app in a timely and cost-effective fashion
- Data quality, especially profile data completeness should be kept to a maximum to enable proper predictions. For instance by requiring a birthdate upon registration, checking income data etc.

References

[1] https://outofmygord.com/2016/09/20/why-our-brains-are-blocking-ads/

[2] https://medium.com/booking-product/personalization-using-machine-learning-from-data-science-to-user-experience-3b1ef5d23ced

[3] https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[4] https://machinelearningmastery.com/k-fold-cross-validation/