RightBound assignment

Eliran Abdoo

April 2022

1 Your initial assumptions – and how you expect the model to behave

My main initial assumption was related to the point you brought up regarding the prediction format - turning it to a multi-class classification problem. A follow up question arose regarding negative example. I ended up thinking about one of two options - either construct a multi-class classifier that utilize only positive samples, and simply tries to predicts the delivery hour, or use the hour as a feature and turn to binary classification, where in inference time we run the prospect against every possible hour, and output the one with the highest probability. I chose the second in order to utilize the negative examples, as well as to overcome the high label imbalance.

2 Explain why you choose this specific algorithm and why it suits the task at hand

Since we're handling a relatively small data-set, 20k samples and highly imbalanced, with a majority of categorical features (as well as absence of options to apply transfer learning), I preferred to go with tree-based algorithms (RandomForestClassifier and XGBoost). They are usually the go-to options for me, especially when considering this type of data and the time constraints. They both work perfectly well for binary and multi-class problems, and adapt their loss functions automatically to logistic-loss and softmax respectively, saving the need for ensembling one-vs-rest constructions.

3 What features did you choose to use and why?

Before considering any feature selection, I had to define which prospect engagements are positive. I adhered to a simple definition where a "delivery" event is positive if it was followed by any non-delivery event. From there, I looked into several things:

- Whether the delivery time on it's own has an effect on the outcome - No real change observed. Hence marking "best hours" alone will yield nothing.
- Whether the number of previous mail deliveries has an effect on the outcome. - Prospects with more than one delivered email have almost double the clicks on average. On the other hand, that group is very small. I still left that feature, as it encompasses some of the prospect's history.
- Whether the time distribution statistics changes in successful deliveries when grouping by different categorical features. Seniority, Size and Position were chosen.

4 If you think more data points can help, suggest which ones do you think are worth collecting and how would you utilize them.

Label imbalance was prominent, so first and foremost getting more positive labels is essential. Oversampling methods like SMOTE might help but the feature space is large, with many categorical values, which might impose a problem. This will also allow usage of regular multiclass labeling. Second, I'd look for balancing the other features, and fill the missing geolocation data, as with larger geographic grasp different working patterns might affect the time people engage with the emails.

5 How do you interpret the results of your suggested model? How would you optimize it if you had more time?

The XGBoost overcomes RandomForest and manages to get around 0.62 ROC-AUC, which is a decent considering the limiting factors (label imbalance, sparse cateogorical features). In terms of the modeling, I'd go for more data understanding and feature engineering, before inspecting different models or hyperparameters tuning. This can also be done via feature importance, which are a nice upside of decision based models. For further understanding I'd even use SHAP - to see which features changes lead to which outcome changes. One other issue I'd explore more is the evaluation, as this is not a regular classification problem, but an ordinal version of it. I've formulated a "cyclic" accuracy metric, which takes into account the fact that we're dealing with periodical classes (i.e. 1:00 AM is close to 23:00 PM). $cyclic\text{-}acc(y, y_pred, peroid) = min((y - y_pred)\%peroid, (y_pred - y)\%peroid)$. Shame it's not differentiable, we could have incorporated it in the loss. Re-assessing all other steps according to that metric could yield better results.

6 Research To Production

Usually I follow the CRISP-DM guidelines:

- Data Understanding - The data structure is very concise, and "home" generated, so it should be straightforward. Main concerns are alot of missing geo-location values, and domain drift overtime. 2 weeks initially, and then ongoing.
- Business Need - Defining the required engagements should also be pretty straightforward. We can convert the events to scores, in case we're interested in more than just a binary engagement type. 1 week.
- Data preparations - there's one source of data, and not much domain expertise to apply. Reducing the broad-domain features (i.e. position, function) should be relatively easy. 1 week.
- Modeling - The hard part here in my opinion. The problem could be modeled in many ways - classification/ordinal regression, forecasting and even sequence classification. I'd drop forecasting since this is a very personalized problem, but would still give a shot to sequence classification as patterns of interactions with users, for example "deliver-deliver-open-deliver-reply" in the correct timing, might lead to interesting results. Couple of months.
- Evaluation - for automatic metric, try incorporating the periodical nature in the metric. Compare the fuzzy with a regular non-fuzzy variations via A/B testing. Hard to asses.
- Deployment - Top 2 considerations for me are scales and system architecture. Since you can't bombard Gmail I guess scale is less of an issue, so I'll stick to the architecture. If it is part of a long pipeline and is feeded constantly, I'd wrap it with message queues to ensure robsutness to failures. Otherwise it can be wrapped as an http/grpc service and serve clients synchronously. 2 weeks to several months, depending on the case and infrastructure readiness.