Our final solution was ensemble of LGB, NN and CatBoost models.

Taking a cursory look at the datasets, it seemed that the features generated from *view_logs* would play a significant role in improving the score. But soon we discovered that most of the features generated from *view_logs* were overfitting the training set. This was because a higher percentage of training data had recent *view_logs* than in the test set. So we focussed more on the feature engineering just on the training set. The features that worked for us were *percentage_clicks_tilldate* per user/app_code/user_app_code. These features alone helped us reach 0.73xx on public LB. Along with these we used some time based features and a couple of features from *view_logs.* We both had slightly different approaches giving similar public LB scores of around 0.75xx. Apart from these common features, we used a few different techniques/features in our individual approaches giving scores of around 0.75xx. Our final submission was ranking average of our individual best models.

Aakash's approach:

- My most powerful features were target based i.e mean of previous clicks for *user_id*, for *app_code* and for *user_id/app_code* combined. Weighted average of above also proved to be important (giving higher weights to more recent records). Other features I created were *time_since_prev_ad, number_of_impressions (for user), sum_of_prev_clicks, impression_time_hour* etc*.*
- I engineered features for *view_logs* separately and then joined based on most recent *server_time* but only one feature "number of views" was helpful while others were overfitting.
- A strong validation strategy was important for this problem as their was intersection in the train and view_logs data. I designed the validation set replicating the test set i.e no intersection between validation set and *view_logs*, this way the CV and LB were close and changes in CV were reflecting on LB.
- Finally I trained 20 lgbm models and 20 fully-connected neural networks (all models having *random state* as the only difference), took arithmetic mean for lgbm models and nn models. And finally took harmonic mean of the two outputs. This gave a public LB score of 0.754x.

Akshay's approach:

- My validation strategy was a simple time-based split.
- Since *view_logs* features were overfitting on training set, I created an extra dataset from *view_logs* dataset: *train_view_logs* (a subset of the original data without an overlap on train dataset)
- I used *train_view_logs* for creating features for training dataset and used original *view_logs* for creating features for test dataset
- *App_code* feature was very important so I decided to encode it in a way so as to capture the hidden relationships between them. I used a word2vec model to convert *app_codes* to 100 dimensional vectors and used them instead of original *app_codes.*
- Apart from these, I created features like *time_since_prev_impression, time_to_next_impression, last_impression_app_code, peak_to_peak_server_time* per user, etc.
- Using these features, I trained 15 LGB models with different seeds and a single CatBoost model. I took a simple average of these models to reach 0.754xx on public LB