TZ Gaming: Optimal Targeting of Mobile Ads

Prof. Hema Yoganarasimhan, Foster School of Business, University of Washington
Prof. Vincent Nijs, Rady School of Management, UCSD

Winter 2021

As a developer of games for mobile devices TZ gaming has achieved strong growth of its customer base. A prominent source of new customers has come from ads displayed through the Vneta ad-network. A mobile-ad network is a technology platform that serves as a broker between (1) app developers (or publishers) looking to sell ad space and (2) a group of advertisers.

App developers sell “impressions”, i.e., a space where an ad can be shown, through the Vneta network to companies such as TZ gaming looking to advertise to app users. Vneta acts as a broker for approximately 50 millions impressions/ads per day.

TZ gaming uses ads to appeal to prospective customers for their games. They generally use short (15 sec) video ads that help to emphasize the dynamic nature of the games. In the past, TZ has been able to, approximately, break-even on ad-spend with Vneta when calculating the benefits that can be directly attributed to ad click-through. TZ, however, believes there are additional, longer-term, benefits from these ads such as brand awareness, etc. that are harder to quantify.

Currently, Vneta provides only very limited targeting of ads to app users but is planning to start offering behavioral targeting to advertisers for a fee. Specifically, two options are under consideration: (a) Provide access to data that advertisers can use to determine which impressions they want to bid on or (b) Advertisers pay Vneta a data science consultancy feel to conduct data-driven targeting on the advertisers behalf.

As Vneta is developing this new business model, it has decided to work with TZ as a partner and has shared behavioral information linked to 115,488 recent impressions used to show TZ ads. Vneta have also provided a set or predictions based on their own (proprietary) algorithm that they intend to use as part of their data science consulting service.

Matt Huateng, the CEO of TZ gaming, is intrigued by the potential for data science to enhance the efficiency of targeted advertising on mobile devices. However, he is not convinced that the consulting services offered by Vneta will be worth the money for future ad campaigns. He has asked you to do some initial analyses on the provided data and compare the generated predictions to Vneta’s recommendations. The following three options need to be evaluated to determine the best path forward.

Options:

1. No targeting (i.e., continue with the current approach)
2. Use predictions from a logistic regression model for ad targeting
3. Use predictions generated by Vneta for ad targeting

The assumptions used for the analysis are as follows:

- Targeting of impressions to consumers covered by the Vneta ad-network to date has been (approximately) random
- Cost per 1,000 video impressions (CPM) is $10
- Conversion to sign-up as a TZ game player after clicking on an ad is 5%
- The expected CLV of customers that sign-up with TZ after clicking on an ad is approximately $25
- The price charged for the data by Vneta is $50K
- The price charged for the data science consulting services by Vneta is $150K

Approach:

- Use the 87,535 rows in the data with "training == 'train'" to estimate a model. Then generate predictions for all 115,488 rows in the dataset
- Options 1-3 should be evaluated *only* on the predictions generated for the 27,953 rows in the data with "training == 'test'". These are the observations that were *not* used to estimate your model
- Extrapolate the cost and benefits for options 1-3 above for an upcoming advertising campaign where TZ will commit to purchase 20-million impressions from Vneta

Although TZ gaming has used RFM for targeting existing customers this approach is not appropriate for prospective customers. Instead, you have decided to use logistic regression. This is a powerful and widely used tool to model consumer response. It is similar to linear regression but the key difference is that the response variable is binary (e.g., click or no-click) rather than continuous. For each impression, the logistic regression model will predict the probability of click-through, which can be used for ad targeting. Like linear regression, you can include both continuous and categorical predictors in your model as explanatory variables.

Matt is eager to assess the value of logistic regression as a method to predict ad click-through and target prospects and has asked you to complete the following analyses.

Part I: Logistic Regression (10 points)

Note: For the following questions, use only the "training" sample of impressions (i.e., 87,535 rows where "training == 'train'"). `click_yes` is defined in the example python code shown below

a. Estimate a logistic regression model using `click_yes` as the response variable and the following as explanatory variables:

`time_fct app mobile_os impua clua ctrua`

The model should predict the probability of `click_yes == "yes"`. Create a new variable called `click_logit` with the predicted click-through probabilities linked to each impression. Use the formula api for `statsmodels` The basic structure of your code to estimate a logistic regression in python should be as follows:

```
import statsmodels.formula.api as smf
tz_gaming["click_yes"] = (tz_gaming["click"] == "yes").astype(int)
lr_mod = smf.glm(
    formula="click_yes ~ time_fct + app + ...",
    family=Binomial(link=logit()),
    data=biden_county,
    data=tz_gaming.query("training == 'train'")
)
lr = lr_mod.fit()
lr.summary()
```

b. Summarize and interpret the logistic regression results. Which of these explanatory variables are statistically significant? Which variables seem to be most "important"? Make sure your model evaluation includes (1) an interpretation of the odds-ratios estimated for the explanatory variables `mobile_os`, `impua`, `clua`, and `ctrua` and (2) an evaluation of the model as a whole.

Use functions from the latest version of the `pyrsm` package to facilitate your analysis (see e.g., https://github.com/vnijs/pyrsm/blob/master/pyrsm/logit.py). You can check the version number of the `pyrsm` package by using `import pyrsm` followed by `pyrsm.__version__`. Your version number should be at least 0.3.4. To install the latest version of the `pyrsm` package use `pip3 install --user pyrsm` from a terminal in the docker container. Tips:

- Use the `or_plot` function from the `pyrsm` package to visualize the Odds-ratios
- Use the `or_ci` function from the `pyrsm` package to get estimates of the odd-ratios

```
import pyrsm as rsm
rsm.or_ci(lr)
```

- Calculate the Pseudo R-squared using the code below. Provide an appropriate interpretation of the returned value

```
print(f"Pseudo rquared {(1 - lr.llf / lr.llnull).round(4)}")
```

c. Estimate a logistic regression model with `click_yes` as the response variable and `imppat`, `clpat`, and `ctrpat` as the only explanatory variable. Make sure to "standardize" the explanatory variables before estimation (see example code below). What is the interpretation of the standardized odds-ratios for the explanatory variables? Tip: Use `or_plot` from the `pyrsm` packages to visualize the odds-ratios and standardized odds-ratios.

```
# scale data by (x - mean(x)) / (2 * sd(x))
from sklearn import preprocessing
X_colnames = tz_gaming.loc[[0], "impup":"ctrpat"].columns
scaler = preprocessing.StandardScaler()
sf = scaler.fit(tz_gaming.query("training == 'train'")[X_colnames])
sf.scale_ = sf.scale_ * 2
tz_std = tz_gaming.copy()
tz_std[X_colnames] = sf.transform(tz_std[X_colnames])
```

d. Some of the variables in the dataset are highly correlated with each other. In particular, `imppat` and `clpat` have a positive correlation of 0.97. Discuss the implications of this (very) high level of collinearity and also different approaches to deal with it. What are the implications for the model and the interpretation of the estimated (standardized) coefficients? As part of your answer, discuss the change in the estimated (standardized) odd-ratio for `imppat` when you remove `clpat` from the model.

Note: To calculate VIF statistics you can use the `vif` function from the `pyrsm` packages. Note that you must use the `lr_mod` model object created above and not the fitted model `lr`

```
rsm.vif(lr_mod)
```

e. Estimate another logistic regression model with `click_yes` as the response variable and `time_fct`, `app`, `imppat`, `clpat`, and `ctrpat` as the explanatory variable. Why are the odds ratios for `imppat`, `clpat`, and `ctrpat` different in the two models? Please be specific and investigate beyond simply stating the statistical problem.

Part II: Decile Analysis of Logistic Regression Results (10 points)

Note: For the following questions, use only the "test" sample of impressions (i.e., 27,953 rows where "training == 'test' ")

a. Assign each impression to a decile based on the predicted probability of click through. Create a new variable `dec_logit` that captures this information. Note: The first decile should have the highest average click-through rate. If not, make sure to "reverse" the decile numbers (i.e., 10 becomes 1, 9 becomes 2, etc.). Please use the `xtile` function from the `pyrsm` package to create the deciles.

b. Create a bar chart of click-through rates per decile (i.e., use `dec_logit` as the x-variable and 'click_yes as the y-variable). Note that the "click through rate" is not the same as the "predicted probability of click." The click through rate captures the proportion of impressions in a given group (e.g., in a decile) that actually resulted in a click.

c. Report the number of impressions, the number of clicks, and the click-through rate for the TZ ad per decile and save this information to a dataframe. Use the name `dec_df_logit` for the new data frame.

Part III: Lift and Gains (5 points)

Note: For the following questions, use only the "test" sample of impressions (i.e., 27,953 rows where "training == 'test' ")

a. Use the dataframe you created in II.c above to generate a table with lift and cumulative lift numbers for each decile
b. Use `seaborn` or `matplotlib` to create a chart showing the cumulative lift per decile. Put cumulative lift on the Y-axis and cumulative proportion of impressions on the X-axis
c. Use the data frame you created in II.c above to generate a table with gains and cumulative gains numbers for each decile
d. Use `seaborn` or `matplotlib` to create a chart showing the cumulative gains per decile along with a (diagonal) reference line to represent the "no model" scenario. Put cumulative gains on the Y-axis and cumulative proportion of impressions on the X-axis

Note: Do not use any specialized packages to construct the lift and gains tables and charts

Part IV: Confusion matrix (5 points)

a. Create a "confusion matrix" based on the predictions from the logistic regression model you estimated above for I.a. Again, use **only** data from the test set here (i.e., "training == 'test' "). Use the financial assumptions mentioned above, and repeated in section V below, to determine an appropriate cut-off (i.e., break-even). Calculate "accuracy" based on the confusion matrix you created (see http://lab.rady.ucsd.edu/sawtooth/RBusinessAnalytics/logit_models.html for an example using R).

   Note: Do not use any specialized packages to construct the confusion matrix

b. Calculate a confusion matrix based on predictions from a logistic regression with `click_yes` as the response variable and `rnd` as the **only** explanatory variable. As before, the model should be estimated on training sample (i.e., "training == 'train' "). Generate predictions for all rows in the data and create the confusion matrix based only on the test set (i.e., "training == 'test' "). Calculate "accuracy" based on the confusion matrix you created.

c. Discuss the similarities and differences between the two confusion matrices. Which model is best, based on the confusion matrix? Provide support for your conclusions.

d. Recalculate the confusion matrices from IV.a and IV.b using 0.5 as the cutoff. Based on these new matrices, discuss again the similarities and differences. Which model is best based on the confusion matrix? Provide support for your conclusions.

Part V: Profitability Analysis (5 points)

Use the following cost information to assess the profitability of using the logistic regression model from I.a for targeting purposes during the upcoming advertising campaign where TZ will purchase 20-million impressions from Vneta:

- Cost per 1,000 video impressions (CPM) is $10
- Conversion to sign-up as a TZ game player after clicking on an ad is 5%
- The expected CLV of customers that sign-up with TZ after clicking on an ad is approximately $25
- The price charged for the data by Vneta is $50K
- The price charged for the data science consulting services by Vneta is $150K

a. Create a new variable `target_logit` that is `True` if the predicted click-through probability is greater than the break-even response rate you calculated in IV.a and FALSE otherwise

b. For the test set (i.e, "training == 'test' "), what is the expected profit (in dollars) and the expected return on marketing expenditures (ROME) if TZ used (1) no targeting, (2) purchased the data from Vneta and used the logistic regression from I.a for targeting, or (3) used Vneta's data science consulting services? You can use the `click_vneta` variable to create a `target_vneta` variable and calculate the expected profit and the expected return on marketing expenditures

   Note: To estimate the performance implications of "no targeting" approach use the predictions from the model you estimated in IV.b

c. Predict the profit and ROME implications for each of the 3 options if TZ purchases 20-million impression for the upcoming ad campaign? Use the results from (b) above to project the performance implications

   Note: The currently available data (+ the click_vneta prediction) are free as part of the partnership between Vneta and TZ-gaming. You should assume, however, that the total cost of the data would be (50K) and that the total cost of the consulting service would be (150K) if used for the 20M impression campaign.

Part VI: Model comparison (10 points)

a. The calculations in V.a through V.c above assume that the predicted probabilities are estimated without error. Calculate the confidence interval for the predictions from the logistic regression model shown below. Now redo the calculations from V.a through V.c, for this model, adjusting for estimation errors. How do your results change?

```
lr_mod = smf.glm(
    formula="click_yes ~ time_fct + app + mobile_os + impua + clua + ctrua",
    family=Binomial(link=logit()),
    data=...,
)
lr = lr_mod.fit()
lr.summary()
pred = rsm.predict_ci(lr, ..., alpha = ...)
```

Create a variable `target_logit_lb` that is `True` if the predicted click-through probability is greater than the break-even response rate and `False` otherwise. Add the columns you need from the "pred" data frame to your data set.

b. You have now estimated 3 different models and also have the predictions from Vneta (see prediction labels below). Compare the models using (1) profit calculations as in V.a through V.c and (2) a gains chart. Discuss which of these models you would recommend to put into production and why.

Prediction labels to use: `click_logit, click_rnd, click_logit_lb, click_vneta`

   Note: For efficiency, you can adapt the `perf_calc` function you created for the Tuango case to do the relevant performance calculations for the different models (i.e., profit, click-through rate, ROME, etc.).

Data description

Information about the sample of 115,488 impressions is in the R dataset tz_gaming.rds (or python dataset tz_gaming.pkl) in the data/R (data/python) directory in the GitLab repo. Each row in the dataset represents an impression that showed a TZ ad. All explanatory variables are created by Vneta based on one month tracking history of users, apps, and ads. The available variables are described below.

- *training* – Dummy variable that splits the dataset into a training ("train") and a test ("test") set
- *inum* – Impression number

- *click* – Click indicator for the TZ ad served in the impression. Equals "yes" if the ad was clicked and "no" otherwise
- *time* – The hour of the day in which the impression occurred (1-24). For example, "2" indicates the impression occurred between 1 am and 2 am
- *time_fct* – Same as *time* but coded as categorical
- *app* – The app in which the impression was shown. Ranges from 1 to 49
- *mobile_os* – Customer's mobile OS
- *id* – Anonymized user ID
- *impup* – Number of past impressions the user has seen in the app
- *clup* – Number of past impressions the user has clicked on in the app
- *ctrup* – Past CTR (Click-Through Rate) (x 100) for the user in the app
- *impua* – Number of past impressions of the TZ ad that the user has seen across all apps
- *clua* – Number of past impressions of the TZ ad that the user has clicked on across all apps
- *ctrua* – Past CTR (x 100) of the TZ ad by the user across all apps
- *imput* – Number of past impressions the user has seen within in the hour
- *clut* – Number of past impressions the user has clicked on in the hour
- *ctrut* – Past CTR (x 100) of the user in the hour
- *imppat* – Number of past impressions that showed the TZ ad in the app in the hour
- *clpat* – Number of past clicks the TZ ad has received in the app in the hour
- *ctrpat* – Past CTR (x 100) of the TZ ad in the app in the hour
- *rnd* – Simulated data from a normal distribution with mean 0 and a standard deviation of 1
- *click_vneta* – Predicted probability of click per impressions generated by Vneta's proprietary machine learning algorithm

The last three letters of a feature name indicate the sources of variation:

- u — denotes user
- t — denotes time
- p — denotes app
- a — denotes ad

Note that there is a clear relationship between the impressions, clicks, and ctr variables within a strata. Specifically: $ctrup = \frac{clup}{impup}$, $ctru = \frac{clu}{impu}$, $ctrut = \frac{clut}{imput}$, and $ctrpat = \frac{clpat}{impat}$.
