Capstone Project Final Report

Partha Solapurkar

1 Introduction

Not every user who creates an account on a website makes a purchase. In this project We analyze anonymized Airbnb user data to glean insights and create models in order to help improve the proportion of users who do make a purchase. We use the Airbnb new user bookings dataset available from `https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings/data`. An initial exploration shows that roughly 60 percent of the users create an account, but do not make a booking. The goal is to work with the marketing team at Airbnb in order to develop strategies to lower this number. This report describes the technical details of the models that were developed.

2 Exploration of the Data

The dataset contains two tables, one table about user information and another table about user activity. The first table, the `train_users` table, contains information about each user account such as the user id, the date on which the account was created, the date of the first booking if the user made a booking, gender, age, signup method, language, marketing affiliate channel, signup app, signup device, browser and the country where the user made their first booking. Most of these are categorical variables. Age is the only numerical variable and there is some timestamp information as well. The second table, the `sessions` table, contains user activity information such as clicks, views and so on. Not every user in the `train_users` table is present in the `sessions` table, and not every user in the `sessions` table is present in the `train_users` table. There are no repeated users in the `train_users` table, but for each user there are multiple entries in the `sessions` table. Every

user in the `sessions` table is a user who signed up in 2014, but `train_users` includes users from 2009 through mid-2014.

3 Preparing the data

The timestamp information was converted into more meaningful features such as year, quarter, month and dayofweek. Five year age buckets were created and the users for whom the age was missing were assigned a separate bucket. The numerical age information was also retained, with all the missing age values set to the average age. All the categorical variables were converted into dummy variables. Only the information about the users from 2014 was retained since the `sessions` table only contains the information from 2014. The information regarding whether a user made a booking or not was separated and the information regarding where a user made their first booking was discarded, since we're concerned with user conversion rather than where the user made their first booking. Some numerical information about clicks, views, the amount of time spent by a user and the amount of user activity generated from the `sessions` table was added to the texttt-train_users table. The dataset prepared for the machine learning algorithms had over 550 features and over 76000 samples. The two classes correspnd to the users who did not make a booking and users who did make a booking. Roughly 62% data was in one class and roughly 38% data was in the other class.

4 Further preparation

The data was randomly separated into training and testing sets, with 80% of the data in the training set and 20% of the data in the testing set. The training data was standardized to have zero mean and unit variance and the testing set was scaled using the scaler that was trained on the training set. Since the training set consists of over 60000 samples, we randomly downsampled it to size 1000, for the purpose of cross-validation. This was done so that the cross-validation process would not run for too long a time or run out of memory. A larger sample size of 10000 users was tested just to be sure, but this made no significant difference.

A number of routines were written to perform the actions that we were

going to perform repeatedly. The routine `perform_grid_search` was written to determine hyperparameters using grid search with fivefold cross validation. Another routine, `return_classifier_stats` was written to return the model statistics such as the accuracy, the log loss, the area under the ROC curve, the precision of the model, the recall of the model, and the F_1-score of the model. The routine `plot_diagnostics` was written to plot the ROC curve, the precision-recall curve and the predicted probability distributions. The last routine, `vary_thresh` was written to display the variation in the precision, the recall, the F_1-score and the accuracy when we move the predicted probability threshold away from 0.5.

5 The models

5.1 Logistic Regression

A logistic regression model, named `logireg`, was tuned by grid search on the type of the penalty and the inverse regularization strength. In retrospect, this turns out to be our best model. The accuracy is near 78% on both the training and the testing set, with no symptoms of overfitting, but possibly with high bias. There are clear directions in which the precision and the recall vary when one varies the threshold away from 0.5.

5.2 Random Forest

A random forest model, named `forest`, was tuned by grid search on the number of trees and the minimum impurity split. This model is grossly overfitting, with over 99% accuracy on the training set and almost 78% accuracy on the testing set. Varying the threshold (on the training set, no cheating!) does not yield any clear directions regarding the changes in the precision and the recall.

5.3 Handling Conflicts

Next we tried taking the average of the logistic regression model and the random forest model, but this did not yield any significant improvement over the logistic regression model. We also tried taking the logical AND and logical OR of the logistic regression model and the random forest model to

handle conflicts in the borderline cases, but no significant improvement was obtained over the logistic regression model.

5.4 Logistic Regression on Steroids

Next we tried a multilayer perceptron with one hidden layer to try to improve the bias of the logistic regression model. Hyperparameter such as the number of logistic regression units in the hidden layer and the regularization parameter were tuned by grid search. The results wa a somewhat better performance on the training set, but no improvement in the performance on the test set.

5.5 Support Vector Machines

We also tried an SVM classifier. The classifier with Gaussian kernel was slightly overfitting with training accuracy roughly 82% and the testing accuracy roughly 78%.

5.6 Feature Engineering

We retained only the top 40 most important features as predicted by the random forest model. This retained about 52% of the cumulation importance of the features. We tried logistic regression on this modified dataset. The results remained similar to the original logistic regression model. Next, we added order two polynomial features $x_i x_j$ where x_i, x_j are among the top 40 features, and again trained a logistic regression classifier. This did not yield any significant improvement either.

6 Suggested Use of the Models

Finally, the best model was the first logistic regression model named `logireg`. We recommend

1. using the `logireg` model in the model_eager_users.ipynb file with a high threshold to identify the eager users with a high precision. The revenue may be improved by prefentially showing more upscale destinations to these users. No promotions are necessary here, since these users are likely to make a booking anyway.

2. using the `logireg` model in the model_hesitant_users.ipynb file with a low threshold to identify the eager users with a high recall. The conversion rate may be improved by offering more promotions to these users, since these users are less likely to make a booking.

7 Further Directions

We may develop further models to predict the country of first booking, rather than just trying to prdict whether the user made a booking. This would be significantly more challenging, since for the countries other than the USA, the proportion of various classes is extremely skewed.