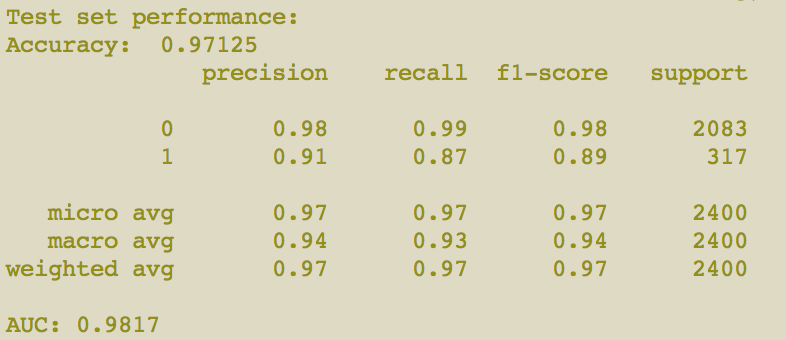
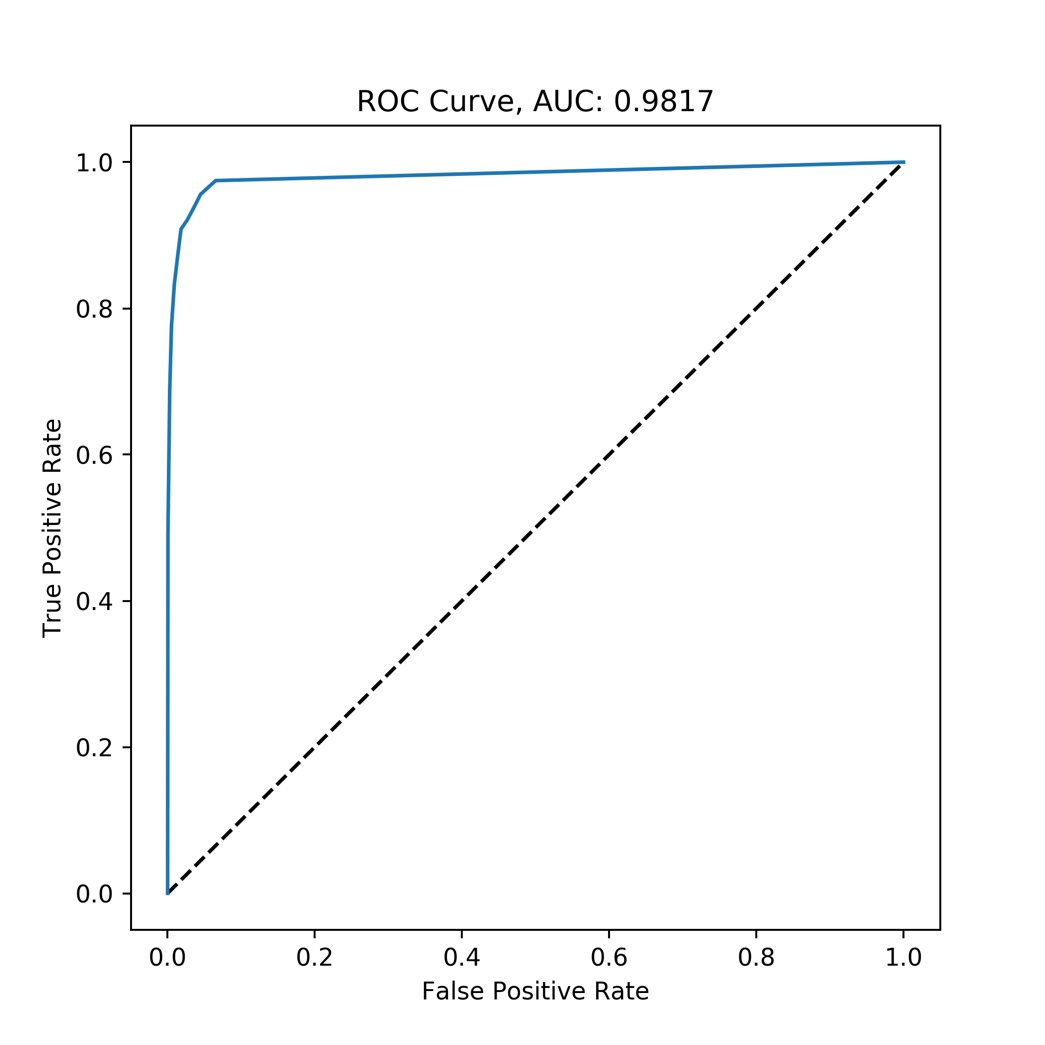
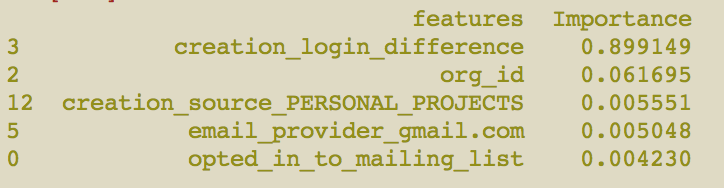
First we clean the table of user information (takehome\_user.csv). We subtract the last\_session\_creation\_time from the creation\_time to get a creation\_login\_difference, which is essentially a proxy for activity on the account. If the creation\_login\_difference is large, you know the user has been using the platform for a long time, whereas if it is zero, it implies the user only created the account and never logged back in. We find the top six most common email providers and make a feature out of those. If a user’s provider is not in the top six, their provider is labeled ‘None.’ We then split the email\_provider feature using one-hot encoding. Next we convert the invited\_by\_user\_id to a binary feature by assigning a one when the user is invited and a zero if not. Next we encode the creation\_source feature using one-hot encoding like we did the email\_provider feature. We then use the take home\_user\_engagement.csv data to find the adopted users as defined in the challenge and join that to our modeling dataset. Finally, we drop the unimportant columns for modeling (object\_id, name).

For modeling, we separate the data into dependent variables (is\_adopted) and independent variables. Then we split the data into an 80/20, training/testing data set for machine learning modeling. We use a Random Forest Classifier because of its ease of use, robustness to different types of data, and capacity for feature selection to model our dataset. Since the number of adopted users is small compared to the total number (~1600/12000, ~13.3%) we use the balanced\_subsample option to adjust class weights inversely proportional to their frequencies for each bootstrap sample of each tree. **Figure** 1 shows the results on the test set after training on the training set. Overall the model performs very well, although recall is a bit low for the positive class (i.e., is\_adopted = 1), indicating that it’s more difficult to accurately predict whether a user is adopted than not. The feature importances are revealing; creation\_login\_difference was by far the most important feature. This might be because we converted that feature to a small unit (seconds) with drastic differences in scale. Regardless, its importance as a proxy for activity and adoption is clear.







**Figure 1.** Performance of our model. Recall suffers for the positive class slightly, but the statistics and ROC curve look good. creation\_login\_difference is by far the most important feature.