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`).

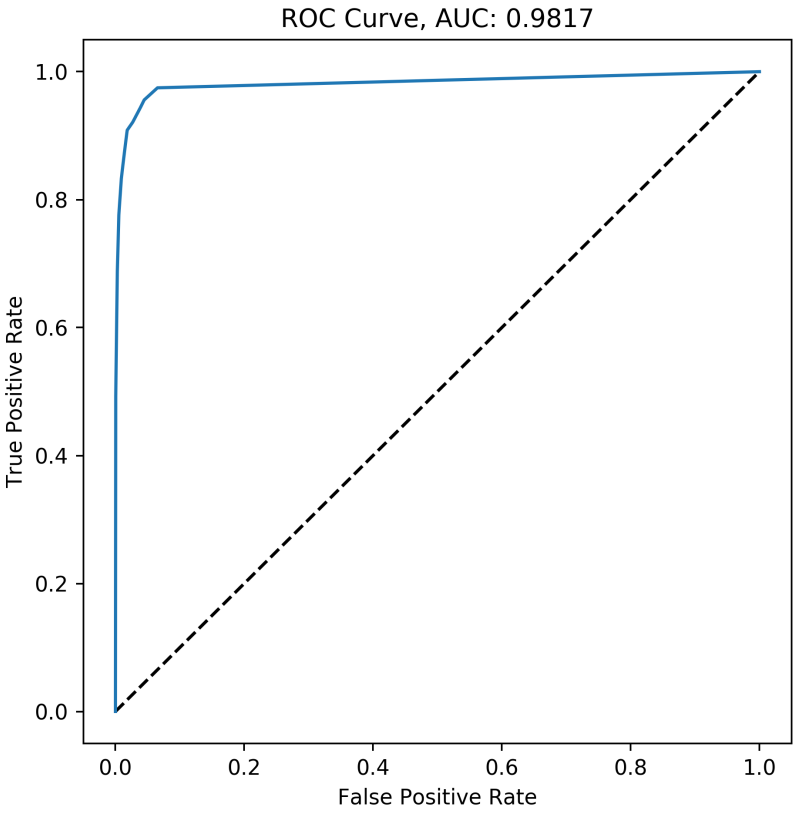
For modeling, we separate the data into dependent variables (`is_adopted`) and independent variables. Then we split the data into an 80/20, training/testing data set for machine learning modeling. We use a Random Forest Classifier because of its ease of use, robustness to different types of data, and 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.

```
Test set performance:
Accuracy:  0.97125
              precision    recall  f1-score   support

           0       0.98      0.99      0.98      2083
           1       0.91      0.87      0.89       317

   micro avg       0.97      0.97      0.97      2400
   macro avg       0.94      0.93      0.94      2400
weighted avg       0.97      0.97      0.97      2400

AUC: 0.9817
```



	features	Importance
3	creation_login_difference	0.899149
2	org_id	0.061695
12	creation_source_PERSONAL_PROJECTS	0.005551
5	email_provider_gmail.com	0.005048
0	opted_in_to_mailing_list	0.004230

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.