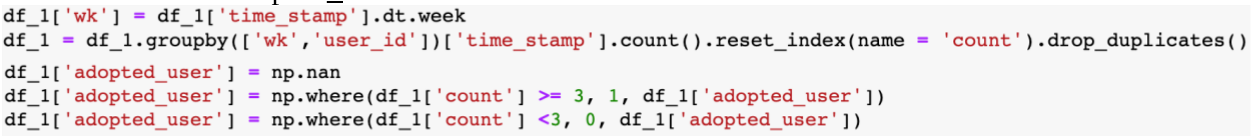
Relax Data Science Challenge

For this Relax Data Science Challenge, my first step was to create an adopted_users column. I did this within the dataframe created by takehome_user_engagement.csv. An adopted user was defined as someone who logs in 3 separate times, on separate days, in a seven-day period. So, after checking that no user had duplicate days on logging in, I used the following code to create an adopted_user column:

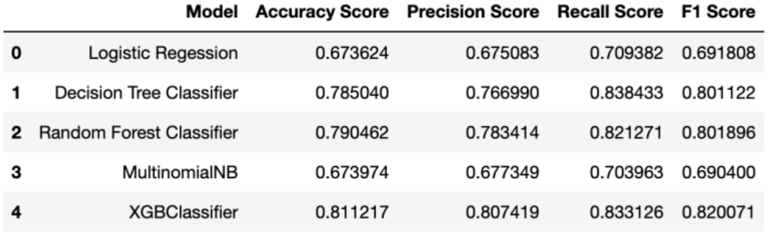


```
df_1['wk'] = df_1['time_stamp'].dt.week
df_1 = df_1.groupby(['wk','user_id'])['time_stamp'].count().reset_index(name = 'count').drop_duplicates()

df_1['adopted_user'] = np.nan
df_1['adopted_user'] = np.where(df_1['count'] >= 3, 1, df_1['adopted_user'])
df_1['adopted_user'] = np.where(df_1['count'] <3, 0, df_1['adopted_user'])
```

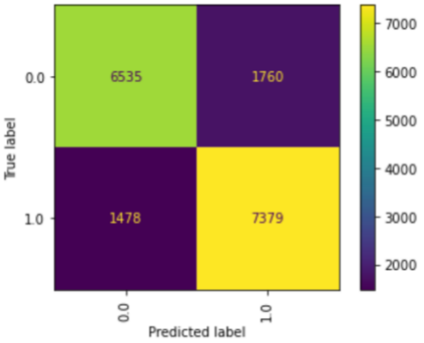
Next, I merged this dataframe with the dataframe created by takehome_users.csv. Once merged, I explored any correlation between adopted_users and the other features. I discovered that one feature, last_session_creation_time had the strongest correlation. After exploration, the last_session_creation_time for adopted_users had a higher mean. Also, more adopted_users had started a session more recently.

I then moved into my predictive model creation. I used sklearn to create multiple models. Using gridsearchcv, I hyperparameter tuned each model. Here are the model results:

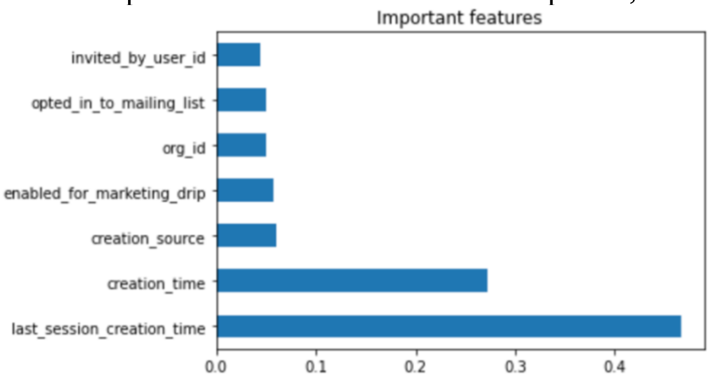


	Model	Accuracy Score	Precision Score	Recall Score	F1 Score
0	Logistic Regession	0.673624	0.675083	0.709382	0.691808
1	Decision Tree Classifier	0.785040	0.766990	0.838433	0.801122
2	Random Forest Classifier	0.790462	0.783414	0.821271	0.801896
3	MultinomialNB	0.673974	0.677349	0.703963	0.690400
4	XGBClassifier	0.811217	0.807419	0.833126	0.820071

The XGBClassifier outscored the other models in every category. Here are the results of the predictions for XGBClassifier:



When the features of importance for this model where explored, here were the results:



The main features for prediction are last_session_creation_time, as predicted earlier, and creation_time.