My steps:
I created the “adopted” class feature using the logins dataset. The steps to creating the feature were rolling window with aggregation resulting in a boolean array. This was converted to a data frame and concatenated to the “users” dataset.
I decided which features to use. The id columns, the email address (unique like id), and the user's names were dropped as they were not informative features.
I looked at the data distribution and checked for duplicated data and missing values. I had to decide what to do with the column that had the id of a person who recommended joining. I decided not to lose this data as it could be very informative and may influence classification. I decided to use 0 instead of missing. This created a multicollinearity problem since the 0 corresponds with people who didn’t have recommendations. I decided to continue with this because the multicollinearity problem existed already and deal with it by selecting a model that will handle it better. The correlated features were “opted into mailing list” and “enabled marketing drip”.
Hot encoding was performed on the categorical features. I had a thought to transform to improve the model's power but we don't have real continuous features.
The next step was to decide how to model the data to classify and predict adopted users.
To choose the algorithms for this classification I was looking for models that fit the most to handle categorical features and also models that could handle the colinearity of some features. This is because we don't have many features and cannot drop any.
I considered LASSO, Ridge Regression, Desition tree, and Logistic Regression.
Because there are not many features and the model can be simple I wanted to compare the models that I will build to a simple classification or regression.
The simple predictor can predict accurately 80 % of the cases. Our models improved this ability up to almost 98%.
I tried Lasso and Desition tree which had very high performance and now could choose between the two without continue checking other models.
The consideration would be what accuracy is expected or needed in this company's project for predicting adopted users. In some cases, even the dummy classifier could be good enough depending on resources of time money, and needs.
Also, the decision tree can give us feature importance.
Last_session_creation_time, 0.643,
creation_t 0.356,
invited_by_user_id_0 0.000,
and also creation_source_SIGNUP_GOOGLE_AUTH , creation_source_PERSONAL_PROJECTS
It looks like the last session is a good predictor also when the account was created.

This means that a company can choose this model if they need more interpretability with a slightly lower accuracy score.
For adoption questions, I would not prefer this option in this case as the important features are the features that the company doesn’t have much control over. I would reconsider this option if there was more data. For better interpretability, there is a need for more features that we can use.
I believe that gathering more behavioral and attitudinal information would be good. This can be obtained by research, and surveys, looking for features that reflect users’ needs, and social networks that influence them. Also using the logins dataset could be useful for creating features like what day of week people log in and some seasonality in general.