Progress Report

Team 42

Dataset

To develop and evaluate our automatic detection algorithm for fake accounts, we selected a pre-constructed public dataset called InstaFake (https://github.com/fcakyon/instafake-dataset). The InstaFake Dataset is comprised of anonymized Instagram user data collected by Fatih Cagatay Akyon and Esat Kalfaoglu and is released in the paper *Intagram Fake and Automated Account Detection* (Akyon 2019). The fake account dataset tath focuses on impersonation in InstaFake contains 200 fake accounts and 994 real accounts – adding up to a total of 1194 data points. Each data point has 8 features including the number of followers/following, biography length, and so on. Please refer to the full feature list from the dataset URL.
We believe that this dataset has covered a wide range of abuse types and variations for the topic we are interested in: fake account detection. For example, the dataset contains some obvious types of fake accounts, including accounts with a very low follower-to-following ratio, accounts that don't have a profile picture, and accounts with few media posts. On the other hand, it also contains some tricky variations, some of which are even hard for humans to identify. For example, there's a fake account that has 500+ followers and frequent media posts. In addition, the dataset even covers an adversarial case we considered in Part 1: using digits or special symbols to substitute username (i.e., username el0n_mu5k is used to impersonate elon_musk). A feature username_digit_count is included in the dataset to target such fake accounts.

Classifier

We have explore a wide variety of classification models during our experiments, including Random Forest, Support Vector Machine, and Extreme Gradient Boosted Tree. Each of these three models has distinct advantages on reducing variance or biases. The performances of all models are listed below:

Extreme Gradient Boosted Tree

Actual \ Pred	Negative	Positve
Negative	194	5
Positive	3	37

Recall: 0,925
Precision: 0.8810

Support Vectore Machine

Actual \ Pred	Negative	Positive
Negative	192	7
Positive	7	33

Recall: 0.825
Precision: 0.825

Random Forest

Actual \ Pred	Negative	Positive
Negative	191	8
Positive	3	37

Recall: 0.925
Precision: 0.822

Extreme Gradient Boosted Tree by far outperforms / matches alternatives on both precision and recall, indicating a strong prediction power that can reduce both variance and biases. It's important to compare the precision and recall of models since it reveals how good can models handle class imbalance which is a huge challenge for our task because the amount of real accounts certainly dwarfs the number of fake accounts.

After fitting our models, we also explored the importance of each features from two perspectives: 1) data variance 2) importance to the model. We first calculated the variance of each feature. The higher the variance is, the more information this feature can provide to separate data points. User_follower_count and User_following_count has the highest two variances. On the other hand, after fitting the Random Forest model, we found out that follower_following_ratio has the highest feature importance (Gini importance reflecting how powerful a feature is on splitting the data). All evidence demonstrated that the follower / following relationship is the most important information on detecting fake accounts.

Successes and lessons learned thus far

Through our part 2 exploration and implementation, we have obtained several successes and learned some lessons. We are happy to find that our machine learning classifier achieves high accuracy, recall, and precision scores. We are confident about its ability to work well in our Discord pipeline. Also, the statistical models we use are highly explainable. Our feature importance analysis could give us insights into how to differentiate fake accounts from real ones.
We also learned some lessons. The first one is that our model is unable to handle tricky fake accounts created by smart spammers. The model fails when a fake account's feature statistics are very similar to a real account. We might need to use extra features or approaches like

similar accounts lookup to improve its performance. The second problem is that our model uses a set of fixed features, which makes it difficult to support new features or new fake accounts variants. This is a tradeoff between using statistical models or more complicated ones. A solution we came up with is to use online learning algorithms to modify the model on the fly.

M3 migration

Here's our plan to incorporate the fake account classifier into our Discord bot. Currently, our Discord bot uses a simple rule-based scoring system to generate suspicious scores for every account. Depending on the suspicious scores, the system will decide whether to pass a user-reported account to the moderator. However, this rule-based scoring system was built on human understanding of the subject and is not reliable or effective. Thus, we plan to substitute the rule-based system with our fake account classifier. We will implement a system to extract features from an account and use our classifier to predict the probability of the account being fake. This probability will be our new automatic flagging that helps filter out malicious reporting.

Citation

F. C. Akyon and M. Esat Kalfaoglu, "Instagram Fake and Automated Account Detection," 2019 Innovations in Intelligent Systems and Applications Conference (ASYU), 2019, pp. 1-7, doi: 10.1109/ASYU48272.2019.8946437.