Detection of Fake Spammer and Genuine Accounts on Instagram

Ellen Dong / Mark Du / Winci Liang / Laurie Ye

Table of contents

01 Introduction & Data Overview

02 Model Development

03 Model Results

04 Discussion

01

Introduction & Data

Problem and Goal

Problem: As artificial intelligence and technology continue to advance, the growth of social media platforms like Instagram has brought an increasing concern regarding the prevalence of fake and spam accounts.

- Distort engagement metrics
- Spread misinformation
- Degrade the overall user experience

Goal: Aim to develop a robust machine learning model to classify Instagram accounts as either authentic or spam based on various account features.

Outcome: Enhance the reliability of online interactions, preserve the integrity of social media analytics, and mitigating the negative consequence of false online behavior. Foster a safe online community for users.

Dataset

Dataset: Instagram Fake Spammer and Genuine Accounts

- Download from Kaggle
- Data Collected using a crawler from March 2019
- Contains 12 variables, 696 observations

Independent Variable	Description
Profile Pic	Indicates whether the user has a profile picture (1 = Yes, 0 = No)
Nums/Length Username	Ratio of the number of numerical characters in the username to its length
Fullname Words	Full name represented in word tokens
Nums/Length Fullname	Ratio of the number of numerical characters in the full name to its length
Name == Username	Indicates whether the username and full name are the same (1 = Yes, 0 = No)
Description Length	Length of the bio in characters
External URL	Indicates whether the account has an external URL (1 = Yes, 0 = No)
Private	Indicates whether the account is private (1 = Yes, 0 = No)
#Posts	Total number of posts made by the account
#Followers	Number of followers the account has
#Follows	Number of accounts the user follows
Dependent Variable	
Fake	Class indicating whether the account is genuine (0) or spam (1)

Data preprocessing: handle missing values and using dummies to encode categorical feature

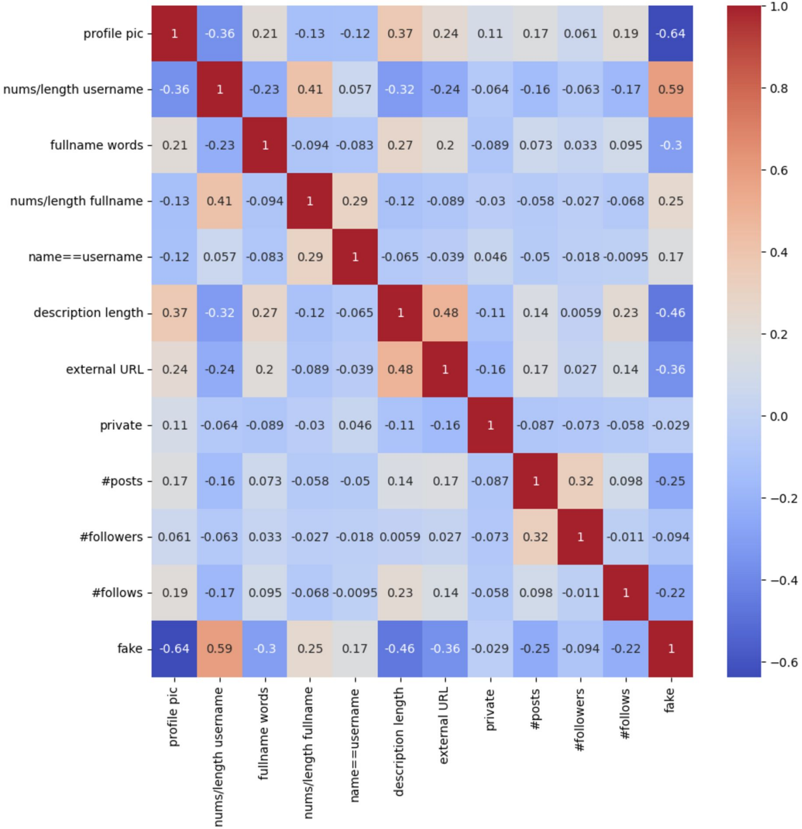
```
# 1) handle missing values
train.fillna(0, inplace=True)
test.fillna(0, inplace=True)

# 2) feature engineering: use dummies to encode categorical features
train = pd.get_dummies(train, drop_first=True)
test = pd.get_dummies(test, drop_first=True)

train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 576 entries, 0 to 575
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   profile pic           576 non-null    int64
 1   nums/length username  576 non-null    float64
 2   fullname words        576 non-null    int64
 3   nums/length fullname  576 non-null    float64
 4   name==username        576 non-null    int64
 5   description length    576 non-null    int64
 6   external URL          576 non-null    int64
 7   private               576 non-null    int64
 8   #posts                576 non-null    int64
 9   #followers            576 non-null    int64
 10  #follows              576 non-null    int64
 11  fake                  576 non-null    int64
dtypes: float64(2), int64(10)
memory usage: 54.1 KB
```

Heatmap: No high correlated features

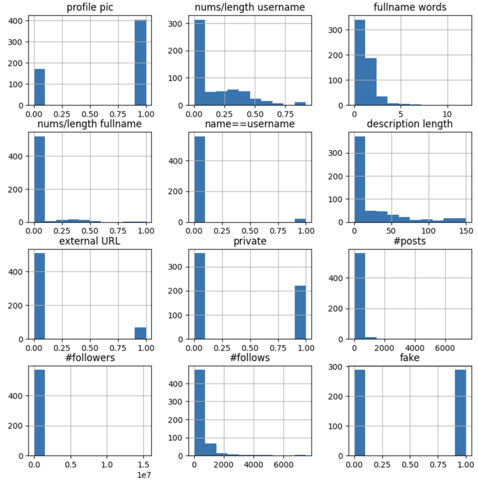


Plot: Top three covariates with “fake”

- Profile pic (negative corr)
- Nums/Length username (positive corr)
- Description Length (negative corr)

```
[7] train.describe()
```

	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
count	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	5.760000e+02	576.000000	576.000000
mean	0.701389	0.163837	1.460069	0.036094	0.034722	22.623264	0.116319	0.381944	107.489583	8.530724e+04	508.381944	0.500000
std	0.458047	0.214096	1.052601	0.125121	0.183234	37.702987	0.320886	0.486285	402.034431	9.101485e+05	917.981239	0.500435
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00	0.000000	0.000000
25%	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	3.900000e+01	57.500000	0.000000
50%	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	9.000000	1.505000e+02	229.500000	0.500000
75%	1.000000	0.310000	2.000000	0.000000	0.000000	34.000000	0.000000	1.000000	81.500000	7.160000e+02	589.500000	1.000000
max	1.000000	0.920000	12.000000	1.000000	1.000000	150.000000	1.000000	1.000000	7389.000000	1.533854e+07	7500.000000	1.000000

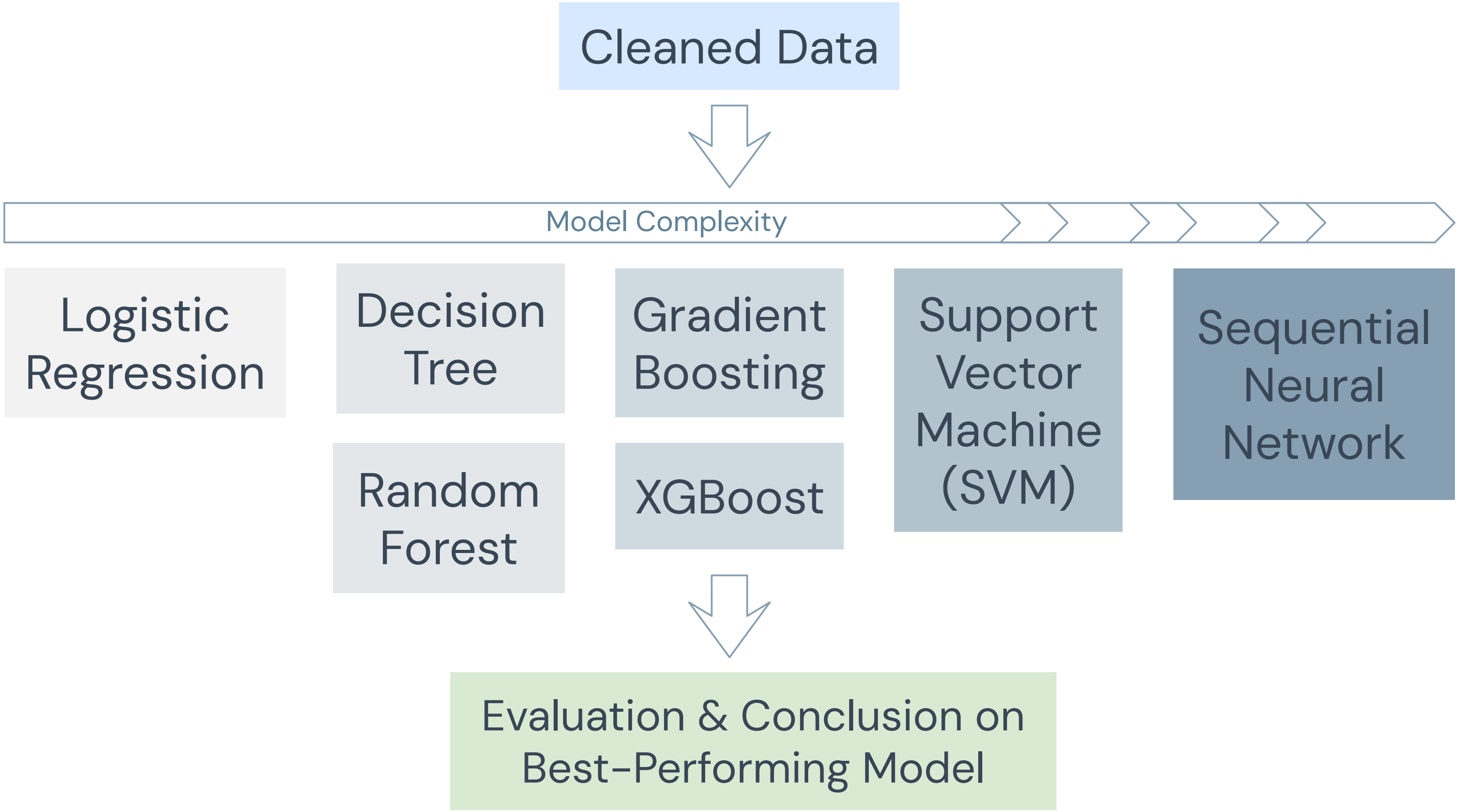


- About 70% of accounts have a profile pic
- Fake accounts are more likely to have usernames with a higher proportion of numbers
  - The average proportion of numbers in usernames is about 0.163, but the max can goes to 0.920
- Fake accounts are more likely to have shorter description
  - 50% of them have no description

02

Model Development

Model Development Diagram



Evaluation Metrics

<u>Our data:</u>
- **50% fake and 50% not fake** Instagram accounts and with not excessively large size, providing a **balanced** scenario.

Accuracy	Precision	ROC-AUC
Provides a straightforward measure of how well the model performs overall	Minimizes false positives (i.e., misclassifying a genuine account as fake) to avoid unnecessary costs	Presents the model's ability to distinguish between the two classes across various decision thresholds

Regression-Based Model Tuning

LR

- Baseline: Default setting
- Hyper-parameter tuning:
  - L1, L2, Elastic Net regularizations
  - Randomized search CV with 5-fold
- Compete results and select the best-performing model within LR models → Only the results from the best-performing is included in final evaluation

Tree-Based Model Tuning

Decision Tree

- Baseline: Default setting
- Tree Pruning:
  - Randomized search CV with 5-fold
  - Cost complexity pruning
- Compete results and select the best-performing model within Decision Tree models

Random Forest

- Baseline: Default setting
- Tree Pruning:
  - Grid search CV with 5-fold
  - Randomized search CV
- Compete results and select the best-performing model within Random Forest models

Boosting Model Tuning

Gradient Boosting

Baseline: Default setting

Hyperparameter Tuning:

- Grid search CV (3-fold)
- Randomized search CV (3-fold)

Compete results and select the best-performing model within Gradient Boosting models

XGBoost

Baseline: Default setting

Hyperparameter Tuning:

- Grid search CV (3-fold)
- Randomized search CV (3-fold)

Compete results and select the best-performing model within XGBoost models

Advanced Non-Linear Model Training

SVM

- Baseline: Default setting
- Hyperparameter Tuning:
  - Grid search CV (3-fold)
  - Randomized search CV (3-fold)
- Compete results and select the best-performing model within SVM models

Sequential Neural Network

- Baseline:
  - Activation layer: ReLU
  - Output layer: Sigmoid
- Tuning:
  - L2 regularization / Dropout / Learning rate (Adam)
- Compete results and select the best-performing model within RNN models

03

Model Results

Model Development Diagram

Model Complexity

Regression-based

Tree-based

Boosting-based

Advanced Non-Linear

Best-Performing Model

LR with default setting

Random Forest with Grid Search CV

XGBoost with Random Search CV

Sequential Neural Network

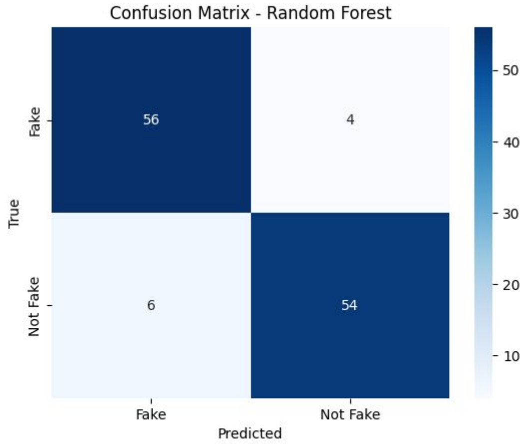
Best Results Overview

Models	Accuracy	Precision	ROC-AUC
Logistic Regression	0.92	0.89	0.97
Random Forest	0.925	0.93	0.99
XGBoost	0.94	0.92	0.98
Sequential Neural Network	0.89	0.91	0.96

Model 1: Random Forest Model

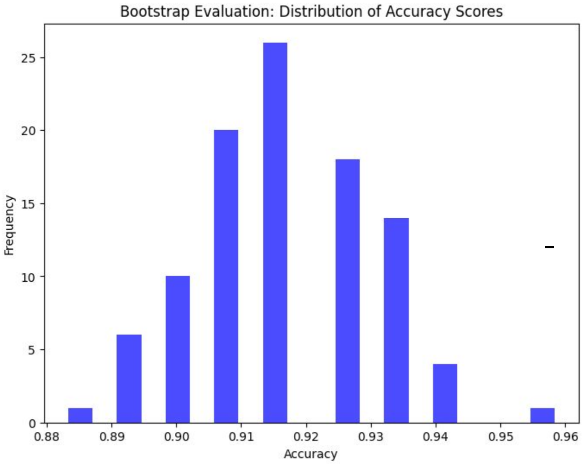
More Evaluation Methods

Confusion Matrix



- The **false positives** and **false negatives** are reasonably low, but there is room for further optimization depending on the specific needs of the application (e.g., reducing false positives if misclassifying genuine accounts as fake is costly).

Bootstrapping

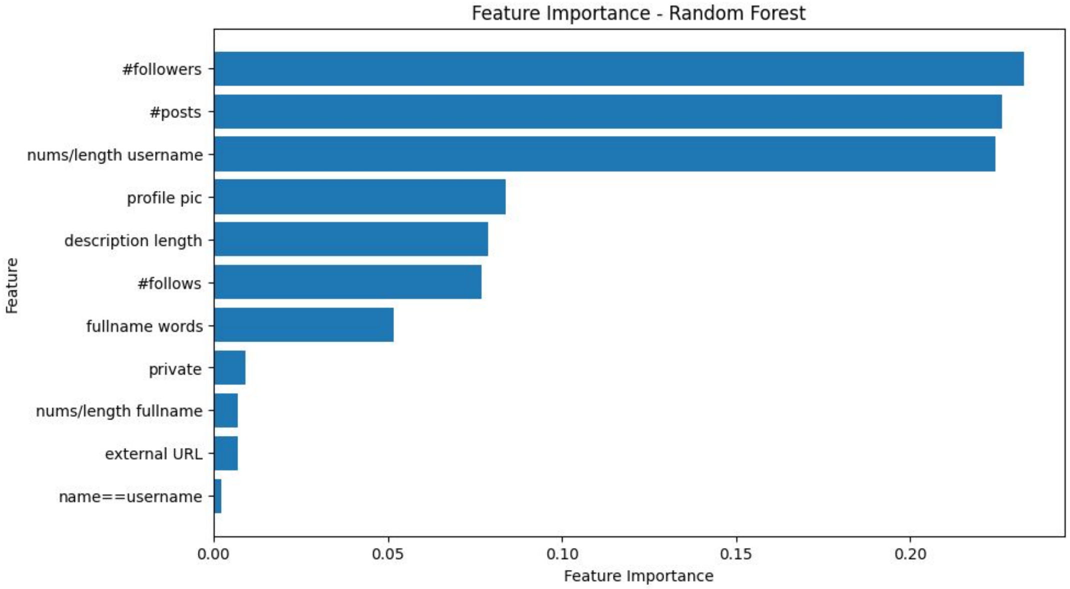


- Performing **well** with an accuracy of around **91-92%**, and this performance is stable as evidenced by the narrow range of the accuracy distribution.

- The **low standard deviation (1.36%)** suggests that the model is not overfitting to specific data samples, but rather generalizes well across different subsets of the data.

Random Forest Model

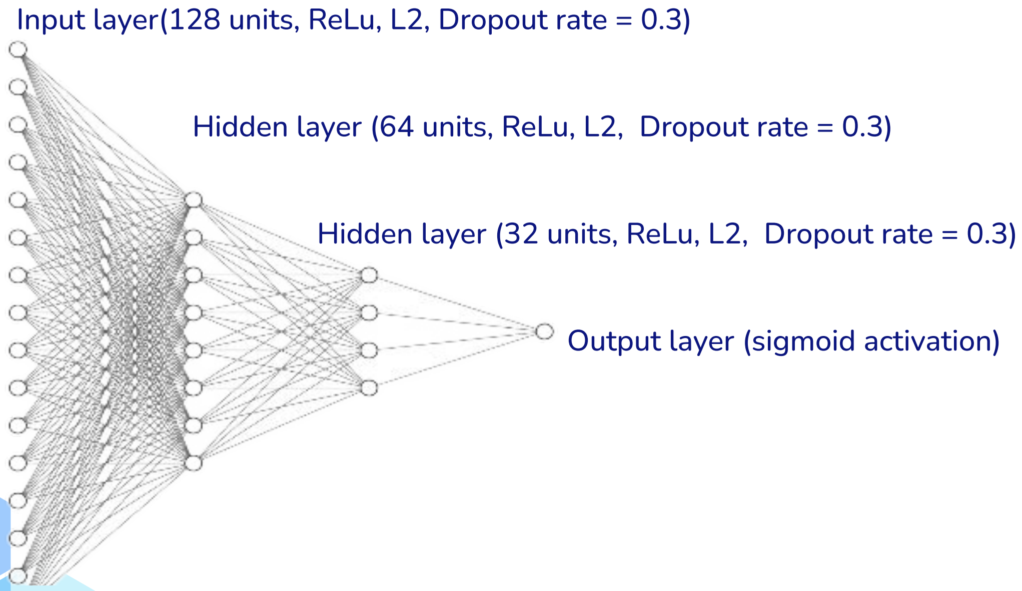
Feature Importance



- **#followers** and **#posts** are the most important features in determining whether an Instagram account is fake or not
- The model relies heavily on typical patterns found in **genuine accounts** (like more followers, posts, and detailed usernames) and less on attributes that might be misleading in fake accounts (such as having an external URL).
- Understanding these feature importances can help in improving the model's focus on critical features and potentially guide strategies to detect fake accounts more effectively.

Model 2: Deep Learning Model

Neural Network: Sequential neural network designed for binary classification.

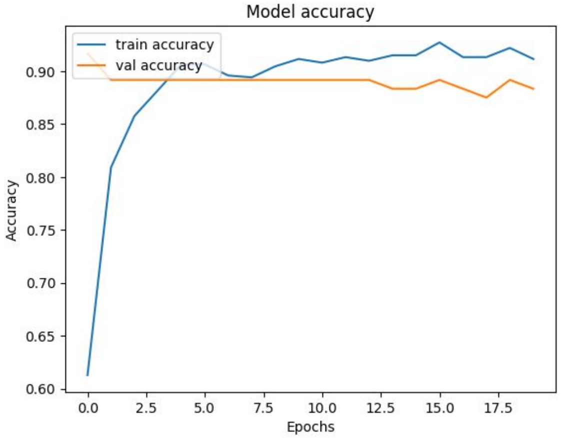


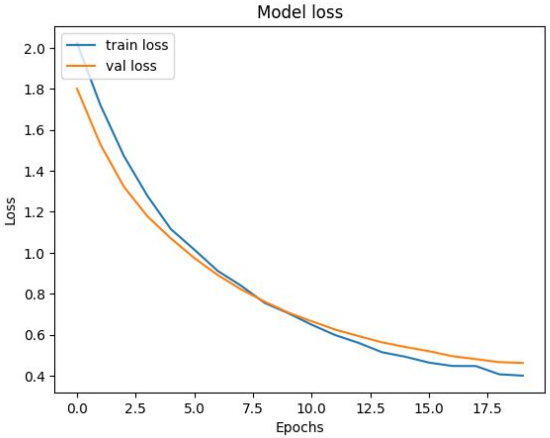
Methods:

- Fine-turning: **Adam Optimizer**
- **Early Stopping**
- Measure Performance: **binary cross-entropy**
- Training: **batch size: 50, epochs: 64**
- Prediction and Evaluation: **Predict on new data and access accuracy**

Deep Learning Model

Neural Network: Sequential neural network designed for binary classification.





Conclusion:

- **The model performs well**, with high accuracy (~90%) and low loss, both in training and validation sets.
- **Minimal overfitting** is observed, as evidenced by small gap between training and validation accuracy/loss.
- **Early improvement**: The model shows quick learning in the first few epochs, and then the performance stabilizes, which is a good sign of convergence.

04

Discussion

Key Insights

Random Forest:

- Balanced accuracy, precision, and interpretability.
- Highlights key distinguishing features like follower count and username characteristics.

Sequential Neural Network:

- Captures deeper, non-linear relationships.
- Potential for better performance with larger datasets.

Performance Comparison:

- Models align with or slightly outperform existing approaches in spam detection (85–95% accuracy).
- Feature engineering (e.g., profile picture presence, username traits) significantly contributed to model success.

Limitations

Feature Availability

Dataset features may not fully capture nuanced spam behaviors.

Inferential

Small dataset (696 samples) limits generalizability.

Conclusion

Summary:

- Developed a machine learning pipeline for classifying Instagram accounts.
- Random Forest was the best performer due to accuracy and interpretability.
- Key features like follower count and username characteristics are critical.

Contributions:

- Provides a scalable framework for spam detection.
- Enhances the reliability of social media analytics.

Future Directions:

- Incorporate additional behavioral features (e.g., interaction patterns).
- Test models on larger datasets to validate scalability and robustness.
- Optimize deep learning models for complex, high-dimensional data.

Thank You !