Instagram Fake Account Detection Project Report

Cpt_S 440 - Artificial Intelligence
Nghia Hoang

Team Members
Kwan Tou To | KwanTou.To@wsu.edu
Nazar Kabylov | Nazar.Kabylov@wsu.edu
Timothy Reidy | Tim.Reidy@wsu.edu
William Walker | William.A.Walker@wsu.edu

Explore our Colab Notebook:
440_Instagram_Spam_Detection
View our Presentation Demo:
Presentation Demo
View our Colab Demo:
Colab Demo
View our Github:
Github

1 Introduction

In the current digital age where social media platforms like Instagram play a pivotal role in our lives, the large amount of fake accounts threatens the user experience and the platform's integrity. According to a study conducted by a research firm Ghost Data, during 2018, as many as 95 million of the 1 billion total Instagram users were fake. [1]

These fake accounts, engaging in activities ranging from spam to scamming, present a significant challenge. Manually distinguishing between real and fake accounts is impractical due to Instagram's extremely large user base, necessitating an automated solution to accurately identify the presence of these fake accounts. Fake accounts not only deteriorate the user experience by spreading misinformation and unwanted spam but also pose security risks and potential financial fraud, undermining both user trust and the platform's advertising revenue.

The primary objective of this project is to develop and evaluate various machine learning algorithms capable of distinguishing between genuine and fake Instagram accounts with high accuracy, addressing the pressing need for automated moderation tools. This study will focus on the effectiveness of binary classifiers in identifying fake accounts based on available metadata and activity patterns.

The data we used was collected from a Kaggle dataset created by Bardiya Bakhshandeh. Bakhshandeh created this dataset using an Instagram web crawler during March 2019, labeling

accounts manually to ensure genuinity. The dataset is balanced to 50% fake and 50% genuine accounts, with pre-set training and testing files, and no noticeable outliers. [2]

The data was well cleaned and already separated into test/train sets when it was uploaded to Kaggle, making our data processing step very straightforward. We first separated each dataset into its features, and its label. The next step was to change the label attribute from the range (0, 1) to (-1, 1), this step is necessary for our custom SVM to train accurately. Our last data processing step was to use the Sci-Kit learn StandardScaler function to standardize all our features.

Our team decided on using multiple different binary classifiers to try to distinguish between the fake and genuine accounts. These classifiers include Naive Bayes, Logistic Regression, Decision Tree, and Support Vector Machines (SVM/SVC). To achieve these goals we used a mixture of hand-made classifiers that we coded from scratch, as well as Sci-Kit learn classifiers to evaluate how well our custom made classifiers performed.

2 Task Objectives

The central objective of the Instagram Fake Account Detection project is to design, develop, and validate machine learning models capable of accurately identifying fake accounts on Instagram. Given the detrimental impact of such accounts on user experience and platform authenticity, our project aims to provide a solution to help enhance Instagram's security protocols and maintain integrity. Our specific goals are:

1. **Model Development**: Construct various binary classification models including Naive Bayes, Logistic Regression, Decision Trees, and Support Vector Machines (SVM). These models will differentiate between genuine and fake Instagram accounts based on patterns derived from user data.
2. **Performance Evaluation**: Assess the performance of each model using precision, recall, F1-score, and other accuracy metrics. The evaluation will focus on the models ability to minimize false positives and false negatives, ensuring a high level of reliability in real world applications.
3. **Model Optimization**: Tune the algorithms using their various hyperparameters to optimize their performance. This involves iterative testing and modification of parameters to achieve optimal accuracy.

Through these objectives, this project seeks to tackle the issues of fake accounts on Instagram, and to observe the feasibility of using an automated solution with minimal user metadata to accurately predict which users are genuine.

3 Technical Approach

To solve our goal of accurately identifying fake users in Instagram, we wanted to test the accuracy of different types of machine learning binary classifiers. The models used include Naive Bayes, Logistic Regression, Support Vector Machines (SVM/SVC), and Decision Trees.

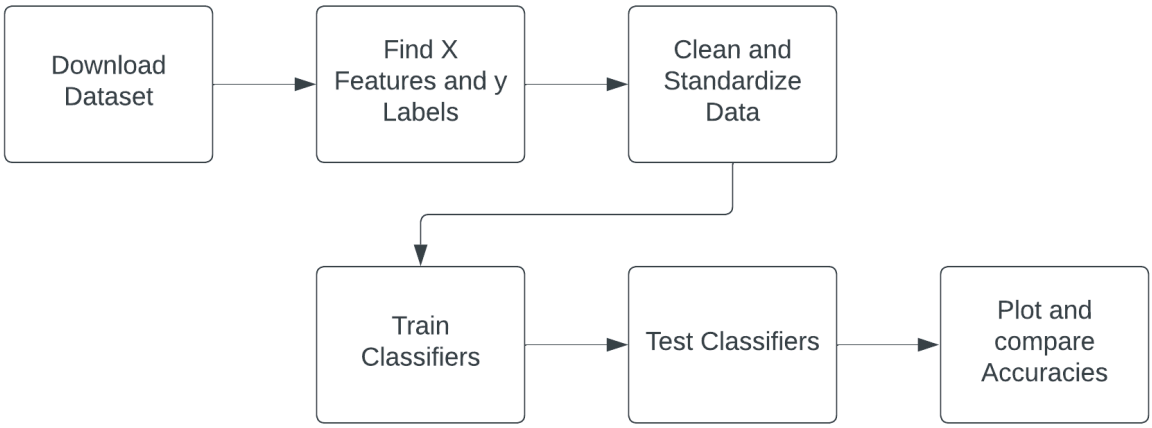
We used the SciKit-learn and Tensorflow modules to train models for each of the classifiers. We then hand coded each of these models to better understand how they function and ensure that our results were reproducible.

For our data, we used a Kaggle dataset created by Bardiya Bakhshandeh. Bakhshandeh created this dataset using an Instagram web crawler during March 2019, labeling accounts manually to ensure genuinity. The dataset is balanced to 50% fake and 50% genuine accounts, with pre-set training and testing files, and no noticeable outliers. [2]

Here is an algorithmic step-by-step outline of our approach:

1. Import the dataset and requirements for our models
2. Process the data by further splitting the test and train sets into feature sets and label sets
3. Clean the data by changing the label set bounds from (0, 1) to (-1, 1).
4. Standardize the data using SciKit-learn's StandardScaler function
5. Create all by-hand and imported classifier models
6. Train classifiers
7. Test classifiers
8. Find optimal classifier parameters
9. Plot and compare the evaluation metrics of each classifier

Figure 3.1. Block Diagram of Technical Approach



This diagram walks through how our team approached the project. A block diagram was chosen because it can quickly and easily communicate all of the specific tasks that need to be completed to finish our project, as well as show how these tasks interact with each other.

4 Evaluation Methodology

To evaluate the performance of our machine learning models for Instagram fake account detection, we employed several standard metrics including accuracy, F1-score, confusion matrix, and ROC AUC score.

We used the `sklearn.metrics` library, specifically the `accuracy_score` function, for the Support Vector Machine (SVM) and Logistic Regression models.

The `accuracy_score` function from `sklearn.metrics` calculates the accuracy of a model's predictions by comparing the predicted labels ($\hat{y}$) with the true labels (y). It works as follows:

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n} 1(\hat{y}_i = y_i)$$

where n is the total number of instances, $\widehat{y}_n$ is the predicted label for the i-th instance, y_n is the true label for the i-th instance, and 1($\widehat{y}_i = y_i$) is an indicator function that equals 1 if $\widehat{y}_i = y_i$ and 0 otherwise.

For the Naive Bayes model, we calculated accuracy using the following equation:

$$Accuracy = 100 \times \frac{\sum_{i=1}^{n} (y_{test_i} = y_{pred_i})}{n}$$

Accuracy for Naive Bayes implementation from scratch measures the overall correctness of the model's predictions and is calculated as:

$$Accuracy = \frac{\{Number\ of\ Correct\ Predictions\}}{\{Total\ Number\ of\ Predictions\}}$$

F1-score: The F1-score is a commonly used evaluation metric that combines precision and recall to provide a single score that balances both measures. Precision represents the proportion of true positive predictions among all positive predictions, while recall represents the proportion of true positive predictions among all actual positive instances. The F1-score is the harmonic mean of precision and recall, ranging from 0 to 1, with higher values indicating better performance. It is particularly useful when dealing with imbalanced datasets, as it takes into account both false positives and false negatives.

Confusion Matrix: A confusion matrix is a table that visualizes the performance of a classification model by presenting the counts of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions. It provides a comprehensive view of how well the model is able to distinguish between different classes. The confusion matrix allows for the calculation of various evaluation metrics, such as accuracy, precision, recall, and specificity. It

helps identify the types of errors the model is making and provides insights into its strengths and weaknesses.

ROC AUC Score: The Receiver Operating Characteristic (ROC) curve is a graphical representation of the performance of a binary classification model at various classification thresholds. It plots the true positive rate (TPR) against the false positive rate (FPR) as the threshold is varied. The Area Under the ROC Curve (AUC) is a scalar value that summarizes the overall performance of the model. The ROC AUC score ranges from 0 to 1, with a score of 0.5 indicating random guessing and a score of 1 representing perfect classification. A higher ROC AUC score indicates better discriminatory power of the model.

5 Results and Discussion

All of our classifiers were trained using our found optimal parameters, and metrics were determined from the prediction output. To compare our models we have found each of their Accuracies, F1 Scores, ROC AUC comparison scores, and Confusion Matrices.

The results of our testing was promising, with our handmade models being very close in metrics to their prebuilt counterparts. We can first examine this by looking at the Accuracy bar graph below. We group the prebuilt and custom classifiers by their type.

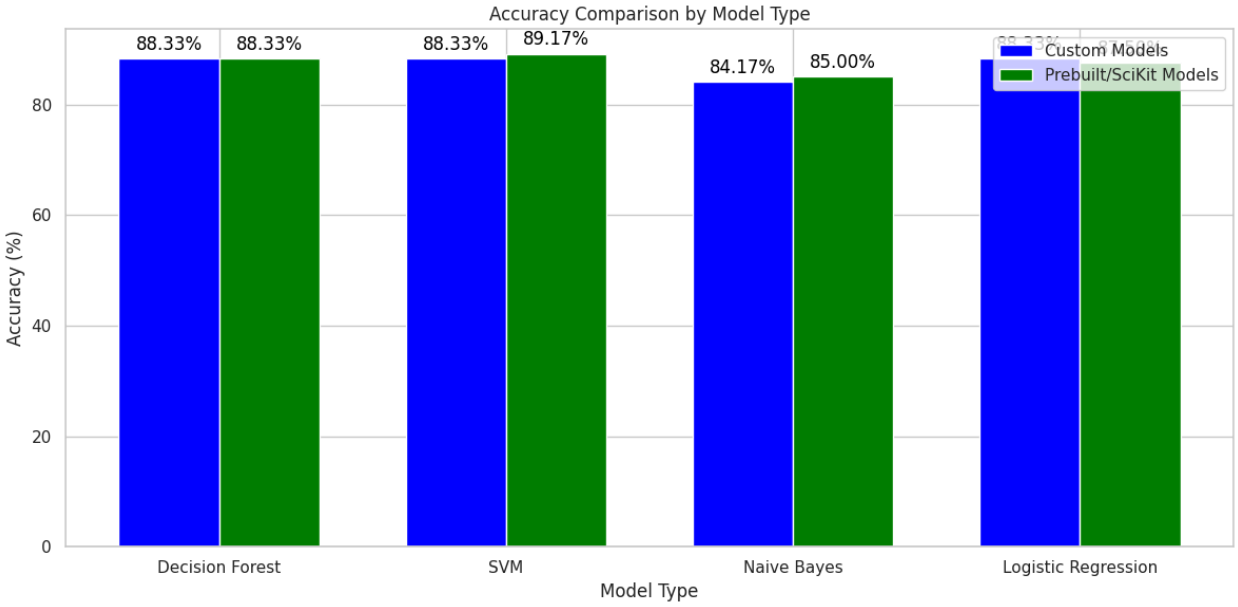
The most accurate of our models was the prebuilt SVM model, by SciKit-Learn, which had an accuracy of 89.17%. The second best models were tied between the SVM, decision tree, and logistic regression models, which all had a maximum accuracy of 88.33%. Our worst performing model was the Naive Bayes model, which still had an impressive 85% accuracy.

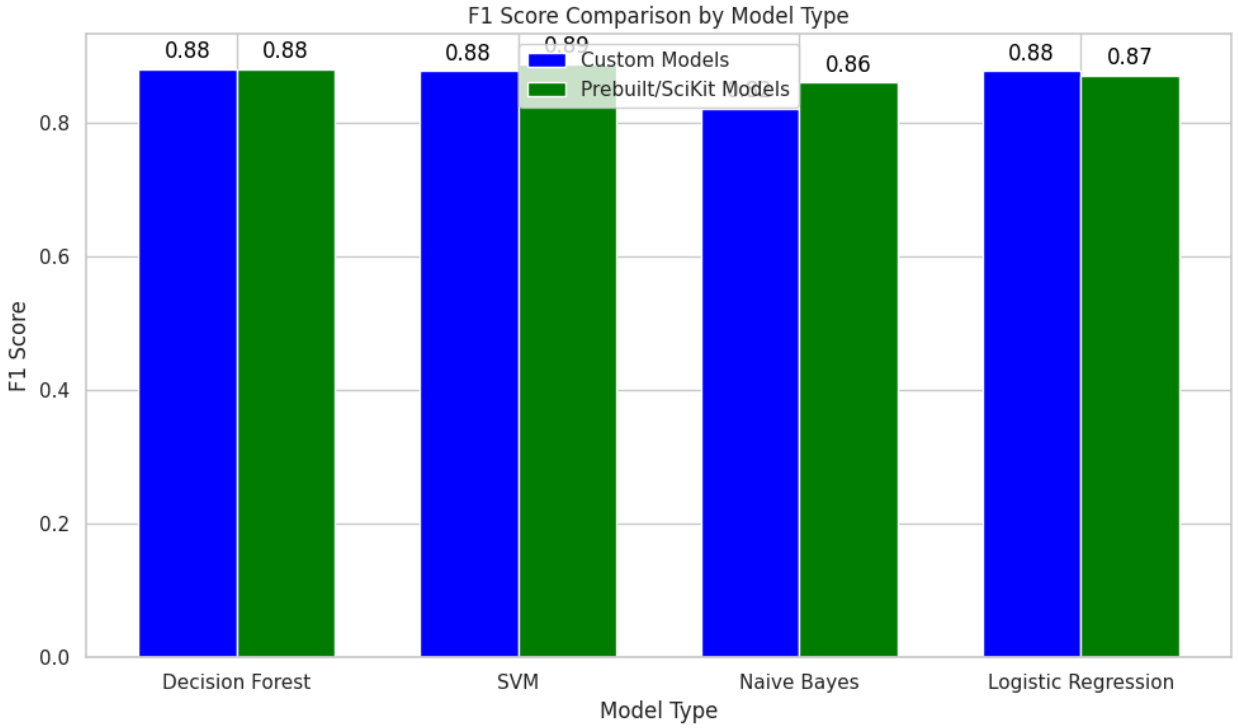
The next metric to observe is the F1 Score. Our model's F1 score bar graph looks extremely close to our accuracy score bar graph. This is because F1 score is impacted by how unbalanced our dataset is, and since the used dataset was perfectly split 50% fake and 50% genuine accounts, the F1 score and the Accuracy score were almost the exact same.

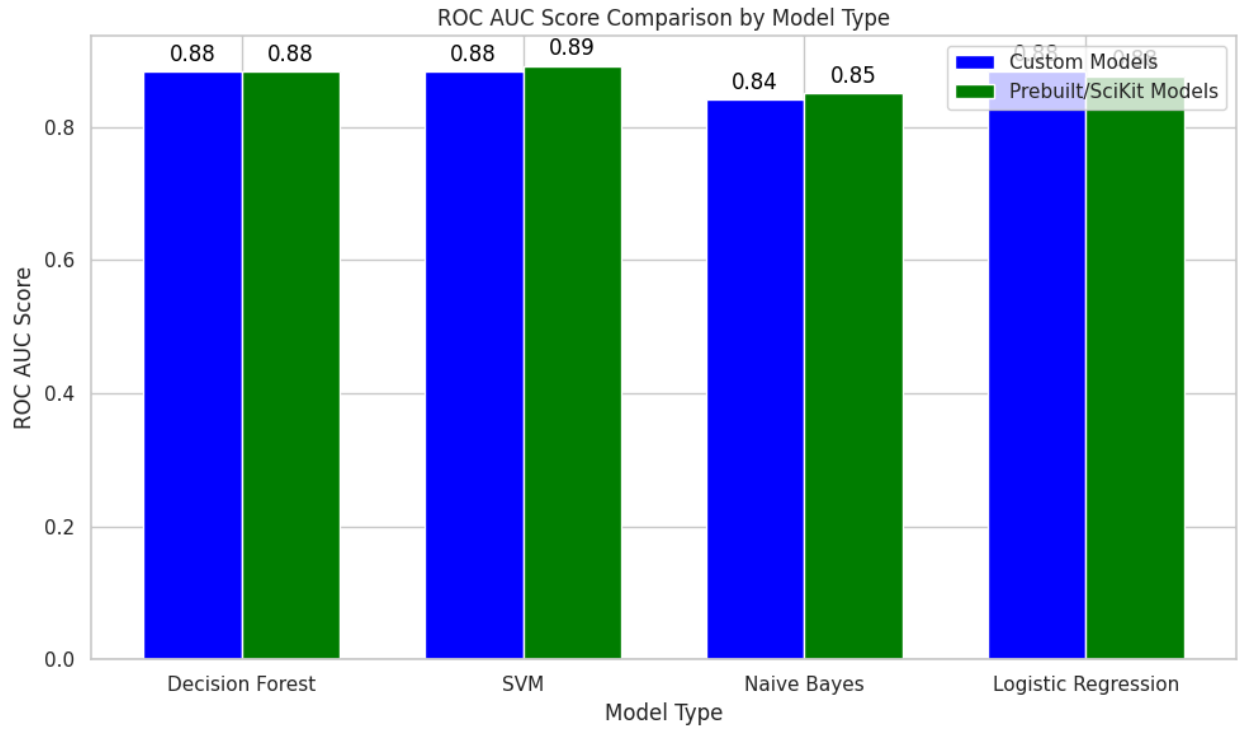
After the F1 Score, we can view the ROC AUC score. Once again this score is extremely similar to the F1 Score and the Accuracy score bar graphs we found above. After further research we believe this to be because our data is so uniform, and because our models predictions are so consistent. This may be because we used the same exact dataset every time, and did not sample it at all.

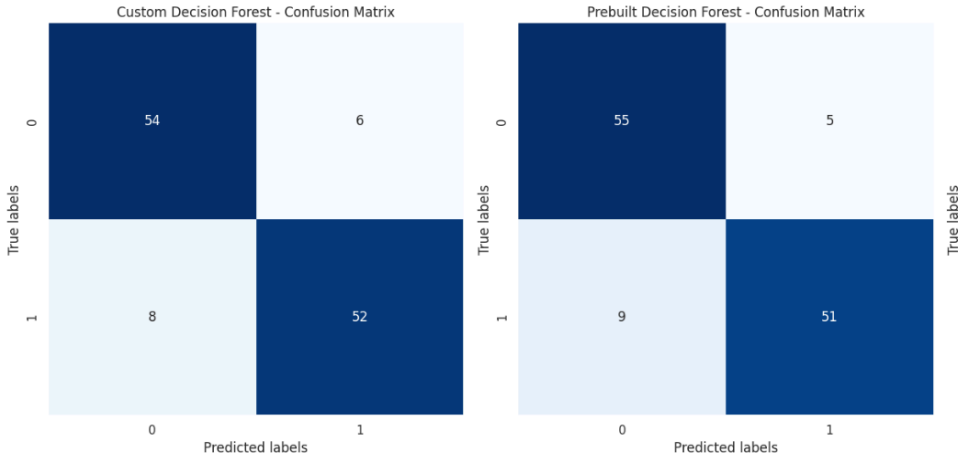
The last metric used by our team was a confusion matrix. By using a confusion matrix we will be able to see how exactly our models classify incorrectly. One note is that the Decision Tree and Naive Bayes models tend to have more false negatives than false positives. This is even truer for our Support Vector Machine models, which had 16 false negatives and only 3 false positives. Our custom Logistical Regression on the other hand, was the opposite and had 14 false positives with only 4 false negatives. The prebuilt model did not have this issue, once again having more false negatives than positives.

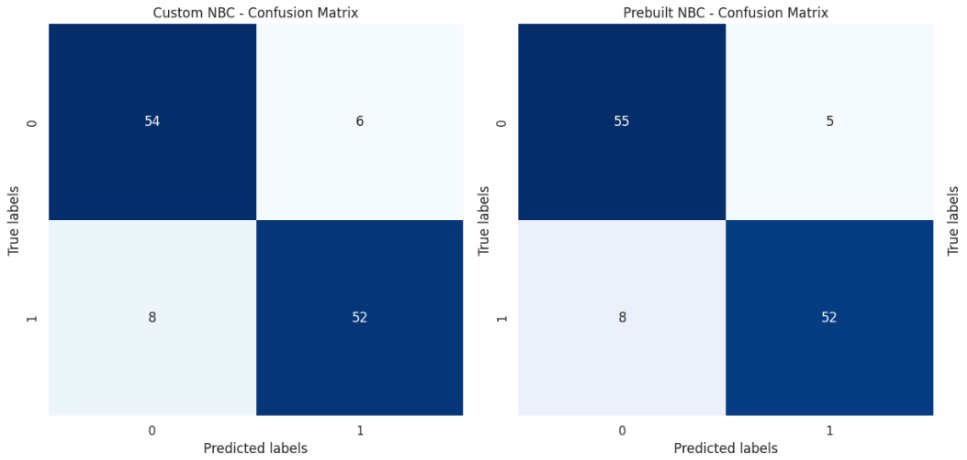
Overall we had a high accuracy efficiency with both our prebuilt and handmade models. We had very uniform metrics with our values being similar for all metrics, implying that our dataset is balanced and our models have consistent predictions. We believe this to be because of our dataset's uniformity, being perfectly split in instances of fake and genuine accounts.

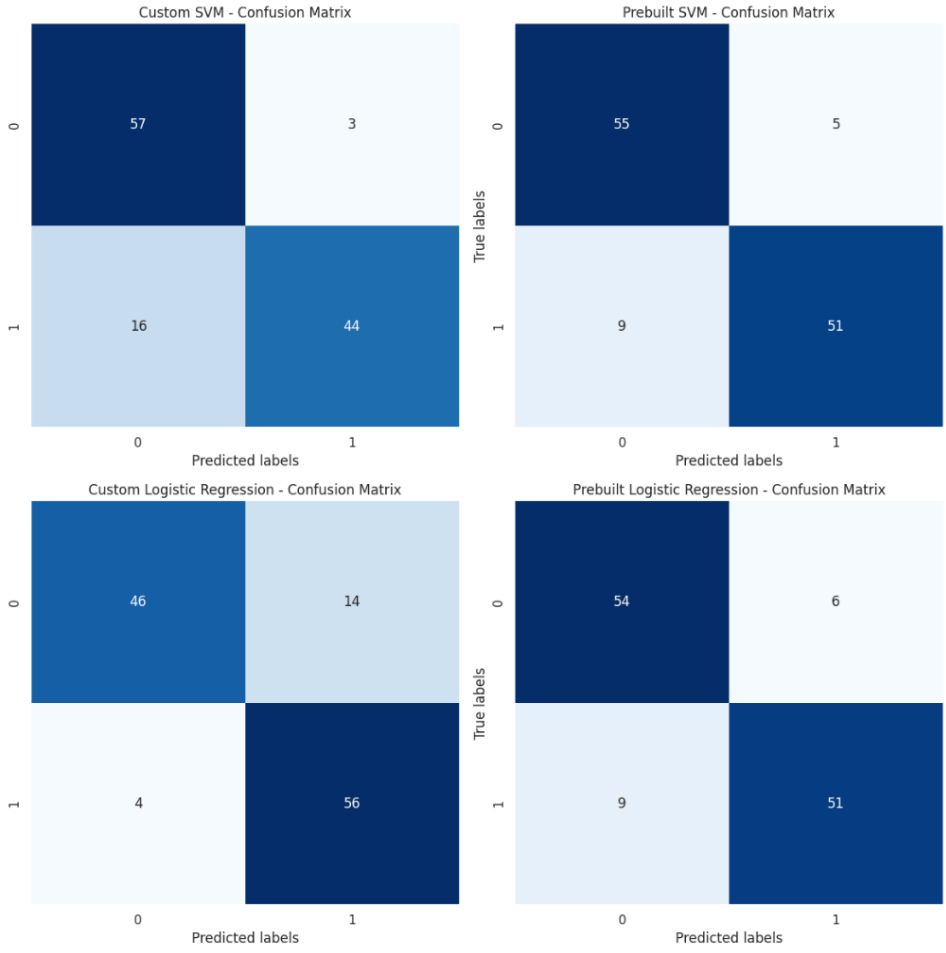





ROC AUC Score Comparison by Model Type
Custom Models
Prebuilt/SciKit Models
0.88
0.88
0.88
0.89
0.84
0.85
ROC AUC Score
0.0
0.2
0.4
0.6
0.8
Decision Forest
SVM
Naive Bayes
Logistic Regression
Model Type


Custom Decision Forest - Confusion Matrix
54
6
8
52
True labels
Predicted labels
Prebuilt Decision Forest - Confusion Matrix
55
5
9
51
True labels
Predicted labels


Custom NBC - Confusion Matrix
54
6
8
52
True labels
Predicted labels
Prebuilt NBC - Confusion Matrix
55
5
8
52
True labels
Predicted labels



6 Future Work

While our initial results are promising, with all models achieving accuracies above 80%, there remains significant room for improvement. Given Instagram's vast user base of nearly 1 billion, even an error rate as low as 10% results in over 100 million users being inaccurately classified. This level of inaccuracy is unacceptable for practical application, emphasizing the need for further refinement of our models.

To enhance the effectiveness and efficiency of our classifiers, we propose the following strategies:

Expand Data Collection:

- **Data Acquisition:** Seek out a larger and more varied dataset to provide a more realistic training environment for our models. By expanding into other social media websites we could access more data

- **Data Sampling:** Implement dynamic sampling during our model training to help improve our models robustness and ability to generalize

Change our Models:

- **Deep Learning:** Implement a deep learning model like a neural network to improve in detecting patterns of fake accounts.
- **Ensemble Methods:** Use our multiple models to create ensemble methods to improve accuracy and reduce misclassification.

Expand to Other Social Medias:

- **Twitter/Facebook:** Similar to Instagram, Twitter and Facebook both have an issue with fake accounts. By using datasets from Twitter and Facebook we can better generalize our models and ensure their effectiveness.

7 References

[1] Albergotti, Reed, and Sarah Kuranda. "Instagram's Growing Bot Problem." *TheInformation*, Lessin Media Company, 18 July 2018, www.theinformation.com/articles/instagrams-growing-bot-problem.

[2] Bakhshandeh, Bardiya. "Instagram fake spammer genuine accounts." *Kaggle*, March 2019, https://www.kaggle.com/datasets/free4ever1/instagram-fake-spammer-genuine-account.

"Getting Started : Tensorflow Decision Forests." *TensorFlow*, www.tensorflow.org/decision_forests/tutorials/beginner_colab. Accessed 25 Apr. 2024.