# Homework 2

Liam Olausson

March 18, 2025

## 1 Feature Construction

To build an effective classifier for distinguishing gender on social media profiles, I extracted a combination of textual and numerical features. The textual fields used were text, description, and name. I employed the following feature extraction techniques:

- TF-IDF
- spaCy Tokenization
- Numerical Features

I first modified the data set to exclude data points that were of the gender brand as these were not important for the classifier that I was creating. Then spacy was used to tokenize text, remove stopwords, lemmatize tokens, and filter out punctuation. This was needed to carry out the TF-IDF on the text, description, and name sections of the dataset. I chose to use these three fields along with the tweet count, the retweet count, and the fav_number in order to help with classification. The reason why I chose name is due to some names having a strong correlation to gender which can help greatly in classification, for a similar reason I chose to use description and text as these can help give more points to classify. For example, if in the description or text there is a mention of the phrase "a mother" there could be a very strong correlation to female. I then added in the numerical data under the assumption that they would be able to further clarify gender as females could have a larger number of tweets, fav_number, and retweets. All of these features are relevant to creating the classifier.

## 2 Description of the Classifier

For this task, I decided to use a logistical regression model to make the classifier. The reason why I chose Logistic Regression is a well-established model for binary classification tasks such as male vs. female. It is interpretable, fast to train, and works well with high-dimensional sparse data such as the TF-IDF vectors. Compared to alternatives like Naïve Bayes, Logistic Regression typically yields better performance on text classification tasks when the features are weighted using TF-IDF. I considered SVM but chose Logistic Regression for its balance between performance and simplicity as well as its far superior speed which was needed for this dataset.

# 3 Evaluation Technique

To evaluate how well the gender classification model performed, I used 10-fold cross-validation. This method divides the dataset into 10 equal-sized "folds," where, for each iteration, 9 folds are used for training the model and 1 fold is used for testing. This is repeated 10 times, with each fold being used exactly once as the test set. This ensures that the model is evaluated across different partitions of the dataset, reducing the risk of overfitting and providing a more robust estimate of performance.

**Evaluation metrics** used:

- Accuracy: Proportion of correct predictions out of total predictions.
- Precision: Ratio of correct positive prediction to total positive predictions.
- Recall: Ratio of correct positive predictions to all positive results.
- F1-Score: The harmonic mean of precision and recall.
- ROC-AUC: Measures the model's ability to distinguish between classes across all thresholds.

These metrics were chosen because they provide a comprehensive view of the classifier's performance beyond simple accuracy. I also included

# 4 Implementation

## 4.1 Preprocessing Steps

I removed all rows where the gender was labeled as brand and dropped missing values in the gender field. To prepare the text data, I used spaCy's en_core_web_sm model for tokenization and lemmatization. Specifically, I removed stopwords and punctuation and kept the lemmas of tokens to normalize different forms of the same word. This was done using a custom tokenizer function based on spaCy's tokeniztion model. I then ran TF-IDF on the text fields in order to prepare of feature extraction. To run the TF-IDF algorithm I used sci-kit's inbuilt TF-IDF function along with my custom function based off of spacy's tokenizer.

## 4.2 Features Used

In order to provide a target dataset to compare my model's performance to I then extracted the gender field from the dataset and mapped it to a binary representaion of 0 for male and 1 for female. This was done so that the model can easily handle comparing the output. I then went on to create a larger array by combining the preprocessed text, names, descriptions, tweet counts, favorite counts, and retweet counts which would then be used to train the model. I kept the retweet count as a dense graph while transforming the retweet and favorite counts to sparse graphs due to the higher prevalance of zeros in these fields.

## 4.3 Classifier Implementation

For the classifier, I used Logistic Regression with a maximum of 5000 iterations, given its robustness in high-dimensional, sparse datasets common in text-based machine learning tasks. While Support Vector Machines were considered, Logistic Regression provided a more computationally efficient solution while still achieving strong performance. The model was trained and evaluated using 10-fold cross-validation, ensuring reliable assessment across different data splits.

## 4.4 Cross-validation and Dataset Partitioning

To evaluate the classifier, I employed 10-fold cross-validation. This method involves partitioning the dataset into 10 equally sized subsets, training on 9 subsets while testing on the remaining one, and repeating this process 10 times. The model was assessed using metrics such as accuracy, precision, recall, F1-score, and ROC-AUC to provide a comprehensive understanding of its performance. Accuracy measured overall correctness, precision captured the ratio of true positive predictions to all positive predictions, recall quantified the ratio of true positives to actual positives, the F1-score balanced precision and recall, and ROC-AUC reflected the model's ability to distinguish between classes across different thresholds. In order to get the right format for the model to train on I used hstack to combine all of the extracted features.

## 4.5 Performance Metrics Calculation

To compute the performance metrics, I utilized the scikit-learn library, specifically the metrics module, which provides convenient functions to calculate accuracy, precision, recall, F1-score, and ROC-AUC. After each fold in the 10-fold cross-validation process, I generated model predictions using the predict() function from the LogisticRegression classifier. For each fold, I passed the true labels (y_test) and predicted labels (y_pred) to the following functions from sklearn.metrics:

- accuracy_score(y_test, y_pred) for calculating accuracy.
- precision_score(y_test, y_pred) for computing precision.
- recall_score(y_test, y_pred) for computing recall.
- f1_score(y_test, y_pred) for computing the harmonic mean of precision and recall.
- roc_auc_score(y_test, y_proba) where y_proba refers to the probability estimates generated by the predict_proba() method of the Logistic Regression model.

The inputs to these functions were the ground truth labels (y_test), predicted labels (y_pred), or predicted probabilities (y_proba[:, 1] for the positive class). The outputs were floating-point values representing each respective metric for that fold. After completing all 10 folds, I aggregated the results by calculating the mean value across all folds for each metric to obtain a robust estimate of the model's overall performance. These averaged metrics were

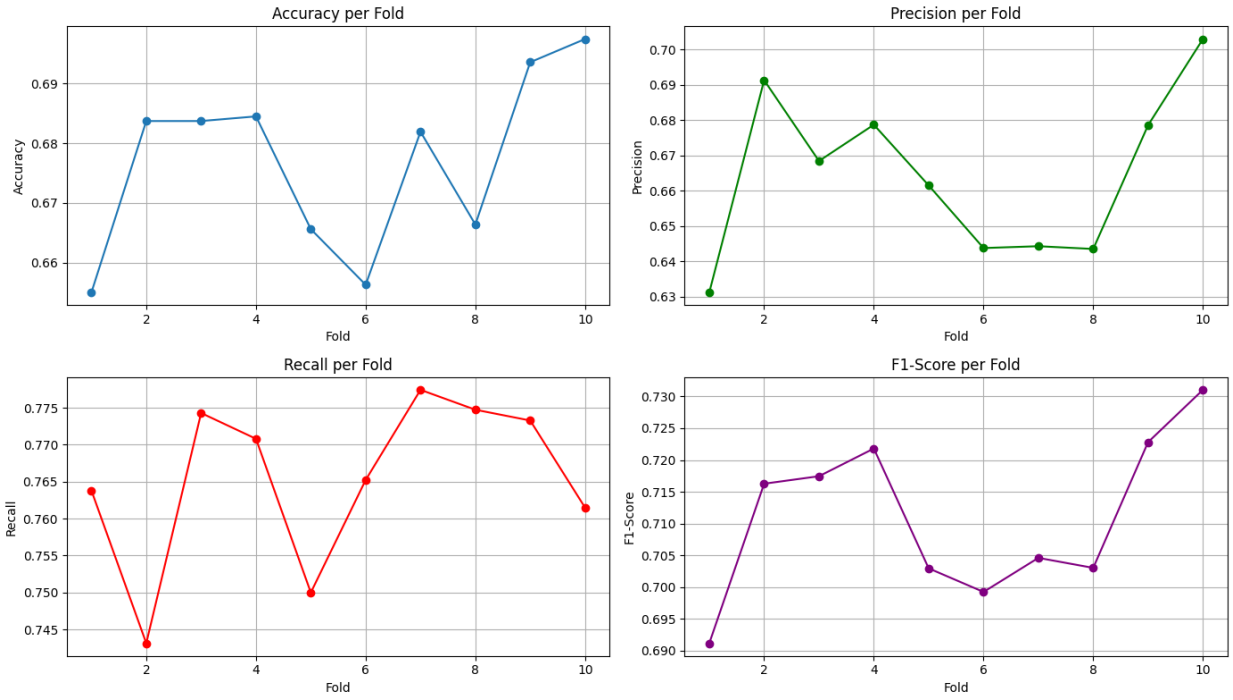


Figure 1: Model Preformance Per Fold

then visualized using matplotlib.pyplot to generate plots such as ROC curves and bar charts for comparison. This approach ensured that the evaluation process was both systematic and reproducible, leveraging standard functions from scikit-learn to guarantee accurate and consistent metric computation.

# 5 Results and Analysis

The final model demonstrated promising results. As shown in Figure 1 and Figure 2, the classifier achieved an average accuracy of 68%, a precision of 66%, and a recall of 77%. These figures suggest that the model is reasonably effective at distinguishing male and female users based on both textual and numerical indicators. The ROC-AUC curves further confirmed that the classifier was able to consistently separate the two classes across different validation folds.

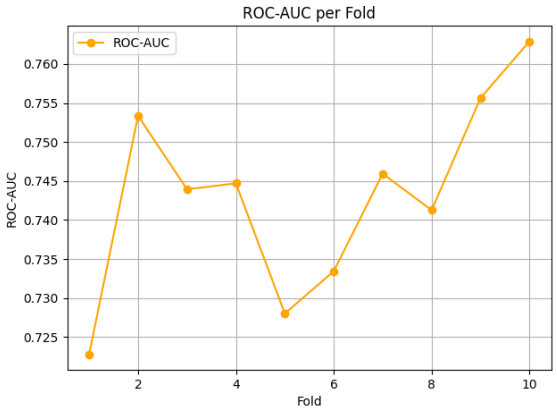


Figure 2: ROC-AUC Per fold

| **Fold** | **Accuracy** | **Precision** | **F1 Score** | **ROC-AUC** |
|---|---|---|---|---|
| 1 | 0.6612 | 0.6376 | 0.6950 | 0.7227 |
| 2 | 0.6915 | 0.6938 | 0.7263 | 0.7547 |
| 3 | 0.6760 | 0.6632 | 0.7093 | 0.7438 |
| 4 | 0.6829 | 0.6734 | 0.7238 | 0.7398 |
| 5 | 0.6633 | 0.6585 | 0.7019 | 0.7262 |
| 6 | 0.6509 | 0.6375 | 0.6968 | 0.7336 |
| 7 | 0.6687 | 0.6305 | 0.6957 | 0.7395 |
| 8 | 0.6687 | 0.6463 | 0.7041 | 0.7474 |
| 9 | 0.6912 | 0.6727 | 0.7240 | 0.7537 |
| 10 | 0.6936 | 0.6983 | 0.7285 | 0.7615 |
| **Average** | 0.6748 | 0.6612 | 0.7105 | 0.7423 |

Table 1: Logistic Regression Performance per Fold

The classifier achieved an average accuracy of 68%, precision of 66%, and recall of 77%. These results indicate a solid performance in distinguishing between male and female social media profiles based on textual and activity features.

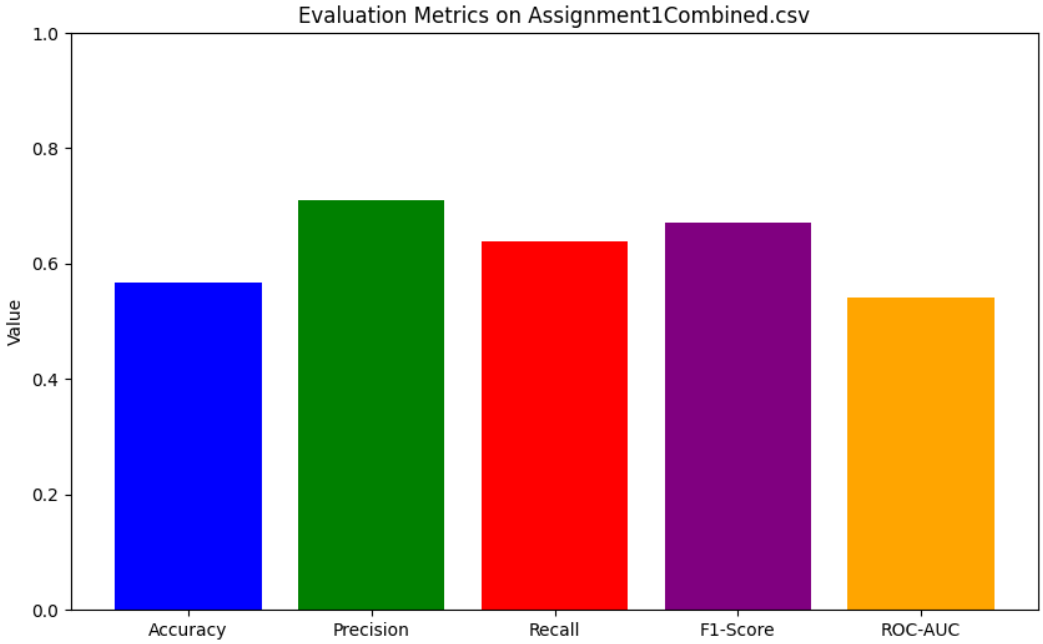


Figure 3: Average Measurements

# 6 Classifing Traditional Data

When applied to the first dataset the model does worse, which makes sense as most of the features that were extracted from the training dataset were not present in the Assignment1 dataset. This makes it much harder for the classifier to work on this new dataset. Overall the Accuracy of this model on the Assignment1 dataset is 57% which is a whole 10% percent below its performance on the initial dataset. The precision was 71% recall is 64%, f1 is 67% and ROC-AUC is 54%.

| **Accuracy** | **Precision** | **Recall** | **F1-score** | **F1-score** |
|---|---|---|---|---|
| 0.57 | 0.71 | 0.64 | 0.67 | 0.54 |

Table 2: Classification Report for Assignment1