CSCI946 Big Data Analytics Assignment 2

Full Name	Student ID	Student Email	Task Log
Robert Mete	5736262	rm700@uowmail.edu.au	Big Data Analytics Lifecycle
Kyuta Yasuda	6588773	ky733@uowmail.edu.au	Dataset Exploration and Preprocessing
Mikael Shahly	8945512	majs774@uowmail.edu.au	Regression Experiments
Thamonwan Nitatwichit	8026300	tn304@uowmail.edu.au	Clustering Experiments
Huu Thien Pham	7794587	htp898@uowmail.edu.au	Dataset Amendment and Assignment Coordination
Syed Eisa Alamgir	8188154	sea994@uowmail.edu.au	Association Rules Experiments
Anas Shahid Raja	8279366	asr968@uowmail.edu.au	Classification Experiments

All members contributed equally.

Table of Contents

Task 1

Phase 1: Discovery

In this phase, the focus will be on understanding the business challenge, exploring the data sources, and formulating initial hypotheses (IH) for potential solutions. This will set the foundation for addressing the problem of distinguishing between human and non-human profiles within Twitter user data to reduce misinformation on social networks. The section is structured into three parts: defining the problem, engaging stakeholders to explore the business domain to develop initial hypotheses, and planning the Big Data Analytics Lifecycle.

Defining the Problem

The objective is to accurately identify and classify Twitter profiles as either human or non-human, a task crucial for maintaining the platform's integrity and mitigating the spread of misinformation, spam, and malicious activities. The rise of bots on social media can distort public opinion, inflate follower counts, and lead to real-world consequences for users and policymakers. Identifying these bots is challenging due to their ability to mimic human behavior and operate on a large scale. Thus, the project aims to develop a scalable and adaptive machine learning model that can continuously identify and mitigate such threats using the twitter_user_data dataset, which includes data from 20,000 random tweets.

Stakeholder Alignment

Stakeholder and project sponsor involvement will be crucial throughout the Big Data Analytics Lifecycle to ensure alignment with project objectives. The team will conduct multiple brainstorming sessions with stakeholders to discuss the nature of the problem, requirements, budget, timeline. It is expected that this project will be conducted in 6 weeks. A structured interview process will then extract detailed requirements, refined into SMART objectives (Specific, Measurable, Achievable, Relevant, Time-bound). Stakeholders will define success criteria to evaluate the project's outcomes. A diverse team with expertise in data science and engineering will be assembled based on these discussions. As the primary data source, the twitter_user_data dataset will be analyzed in detail. Once objectives and data sources are agreed upon, both teams will develop initial hypotheses, such as non-human profiles having more followees than followers or exhibiting unusual activity patterns.

Planning

1. Discovery (Week 1)

The first week will involve understanding the project scope, objectives, and data requirements. Initial meetings will define project goals, identify primary data sources, and outline potential challenges. Deliverables will include a project scope document and an assessment of data sources.

2. Data Preparation (Week 2)

In the second week, the team will focus on cleansing and preparing the data. This will involve acquiring data, removing inconsistencies, handling missing values, and consolidating the dataset into a single format. A prepared dataset and a data quality report will be the key deliverables.

3. Model Planning (Week 3)

Week three will involve outlining the modeling approach and selecting suitable algorithms. The team will review existing models relevant to classifying Twitter profiles and decide on algorithms based on data characteristics and project goals. A model plan, including features and metrics, will be developed, along with proof-of-concept models.

4. Model Building (Weeks 4 and 5)

During weeks four and five, the focus will be on developing and refining predictive models. The models will be trained on the prepared dataset, with parameters adjusted for optimal performance. The team will validate the models using a subset of the data to ensure accuracy. The final deliverables will include trained models and a model validation report.

5. Communicating the Results (End of Week 5)

By the end of week five, a detailed report and presentation will be prepared to summarize the modeling efforts, findings, and business implications. A review meeting will be conducted with stakeholders to gather feedback and discuss the results. The deliverables will be a comprehensive presentation, report, and a document capturing stakeholder inputs.

6. Model Deployment (Week 6)

In the final week, the model will be deployed into a production environment. The team will finalize the model based on feedback and work with IT and data engineering teams to integrate it into business processes. A deployment and monitoring plan will be established to ensure ongoing performance evaluation.

Phase 2: Data Preparation

This phase involves learning about the data, conditioning it for analysis, transforming it into a suitable format, and visualizing it for insights.

1. Data Conditioning

A detailed analysis of the dataset will be performed to uncover patterns, detect anomalies, and understand interrelationships. Initial inspections using methods like df.info() and df.head() will help ascertain the dataset's composition and highlight areas with missing data. Columns with excessive missing values (over 90%), such as gender_gold and profile_yn_gold, will be removed. For columns with less critical missing data like description and tweet_location, placeholder values will be used to maintain dataset integrity.

2. Data Transformation

Numerical attributes will be normalized using the StandardScaler, and categorical attributes will be processed through one-hot and label encoding methods. Additionally, hex color values will be transformed into RGB format to simplify analysis. Text data is processed through cleaning and normalization steps such as lowercasing, punctuation removal, tokenization, and lemmatization. This preprocessed text is then transformed into numerical vectors using TfidfVectorizer, capturing the significance of words in the dataset.

3. Data Visualization

Data visualization provides essential insights into the dataset's structure and relationships among features. Tools like histograms, scatter plots, and heatmaps are employed to display the distribution of numeric data and its correlation with the target variable. Count plots are used for categorical features to show the frequency of different categories, while scatter plots illustrate the interactions between numeric and categorical data. This visualization process helps identify patterns and anomalies, guiding feature engineering and model planning in the subsequent phases.

Phase 3: Model Planning

This phase involves feature engineering, variable selection, and model selection to identify the most relevant features and algorithms for classifying Twitter profiles.

1. Feature Engineering

Feature engineering is critical for creating new features that capture user behaviors over time, thereby improving the classification of human and non-human profiles. New features like account_age, tweets_per_day, and retweets_per_day will be created to capture user behaviors over time. Extracting the year from date fields will generate features such as profile_created_year, enhancing the model's ability to distinguish between different profile types.

2. Variable Selection

Variable selection is performed using various techniques to identify the most relevant features and enhance model accuracy. Numerical features are analyzed using Pearson correlation coefficients to detect and eliminate highly correlated features, thereby avoiding multicollinearity. Categorical features are evaluated with chi-squared tests to determine their significance concerning the target variable. Additionally, some selected models in the upcoming section, such as linear regression and ensemble tree-based models, have intrinsic feature selection capabilities. These models will be used to guide the process of selecting the most influential features, ensuring the model focuses on variables that provide the most predictive power

3. Model Selection

Selecting the appropriate models is crucial for accurately classifying Twitter profiles. Depending on the dataset's nature and the problem's complexity, various models are explored.

Clustering and dimensionality reduction techniques like KMeans and DBSCAN Clustering are used to identify natural groupings within the data, potentially revealing patterns distinguishing human from non-human profiles. UMAP (Uniform Manifold Approximation and Projection) is applied to visualize data distribution, aiding in feature selection and model training.

For classification tasks, XGBoost is chosen for its robustness and ability to handle complex relationships within the data. Gaussian Naive Bayes is also included as a baseline model due to its simplicity and effectiveness, particularly when dealing with text data. For regression tasks, the Gradient Boosting Regressor is employed to predict continuous outcomes and offering insights into feature importance. Ridge Regression is also used to explore relationships between fav_number and other features, providing a deeper understanding of how numerical and encoded categorical features influence predictions.

Phase 4: Model Building

This phase involves developing training and testing datasets, choosing appropriate evaluation metrics, and tuning parameters for model optimization.

1. Developing Datasets for Training and Testing

The preprocessed dataset will be split into training and testing sets using stratified splitting to maintain balanced target variable distribution. The training set (70-80%) will be used to train the model, and the testing set (20-30%) will evaluate performance.

2. Metrics Used and Their Justification

Different metrics will be used based on the problem type. For classification tasks, accuracy, precision, recall, and F1-score will be employed to provide a comprehensive evaluation. For regression tasks, metrics like mean squared error (MSE) and R^2 score will be used to measure prediction accuracy and explanatory power.

3. Parameter Tuning

Parameter tuning is essential for enhancing model performance. Hyperparameters for models like XGBoost (e.g., max_depth, learning rate) and Gradient Boosting Regressor (n_estimators, learning_rate) will be fine-tuned using grid search and cross-validation to improve accuracy and robustness.

Phase 5: Communicating Results

Effectively communicating the results to stakeholders is crucial. The objective is to translate complex model outputs into actionable insights that align with the project's objectives. A comprehensive report and presentation will summarize the models used, their performance metrics, and key findings. Visual tools like confusion matrices, ROC curves, and feature importance charts will make the results more accessible. An open discussion will be facilitated to gather feedback, address concerns, and explore next steps.

Phase 6: Operationalizing

Operationalizing the model involves integrating it into a production environment for real-time or batch predictions.

1. Integration

The model will be deployed on a scalable platform, such as AWS or an internal server, with APIs or web services set up for real-time predictions. Automated data preprocessing will ensure incoming data is correctly transformed.

2. Monitoring

Continuous monitoring will track key performance metrics to maintain model accuracy and detect data or model drift. Alerts and dashboards will provide real-time feedback, enabling quick responses to performance issues.

3. Actionable Use

The model's outputs will be integrated into business workflows to guide actions, such as flagging non-human profiles for review. Dashboards and reports will provide stakeholders with easy access to insights for data-driven decision-making.

Tasks 2 & 3

Exploratory Data Analysis (EDA) and Data Preprocessing

Dataset Overview

Starting with fetching the dataset twitter_user_data.csv, which consists of Twitter users' profiles and tweet information. A quick look at the dataset revealed several features such as user demographics, tweet-related statistics, profile aesthetics (e.g., link_color and sidebar_color), and text data (e.g., description and text of tweets). The dataset had 20,050 rows, each representing a unique user, and included features like gender, tweet_count, retweet_count, fav_number, and created (Twitter User Data).

EDA

1. Overview of the Dataset

The dataset consists of Twitter user information with 20,050 rows and 26 columns. Each row represents an individual user and includes a variety of features such as `gender`, `description`, `tweet_count`, `retweet_count`, and `fav_number`. The goal of the analysis is to explore the data, clean it, and prepare it for a classification task that differentiates between human (male, female) and non-human (brand) accounts. Some features were explored to discover their distribution by gender for consideration whether to eliminate or retain them.

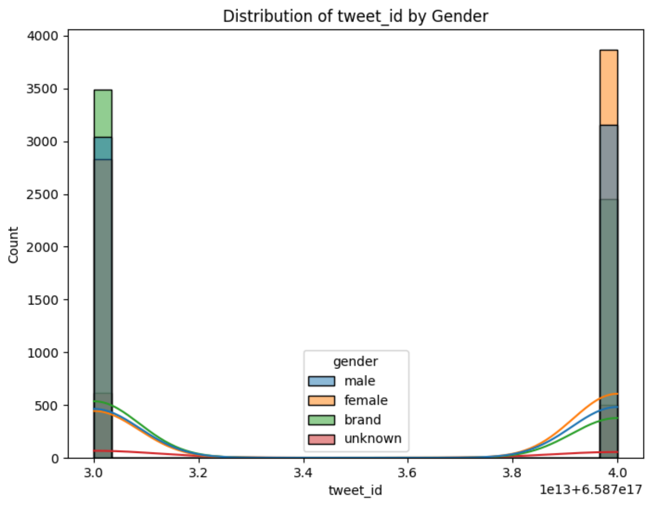


Figure 1: The distribution of tweet_id by gender

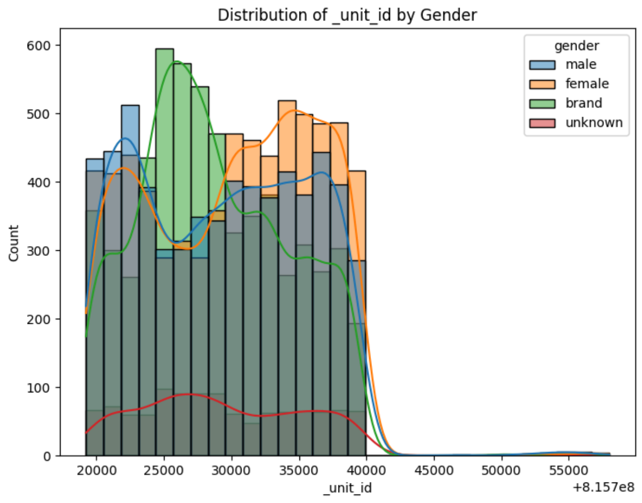


Figure 2: The distribution of_unit_id by gender

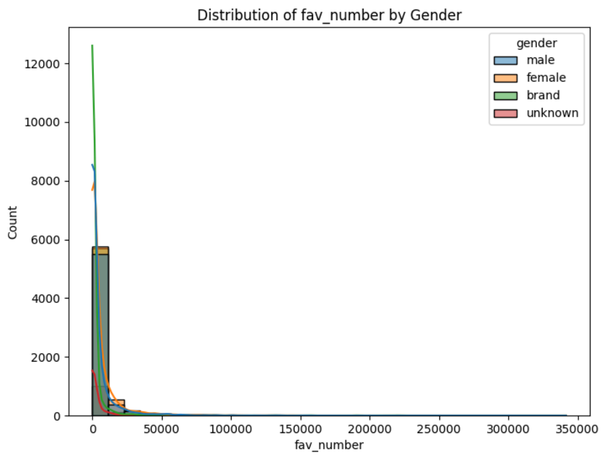


Figure 3: The distribution fav_number by gender

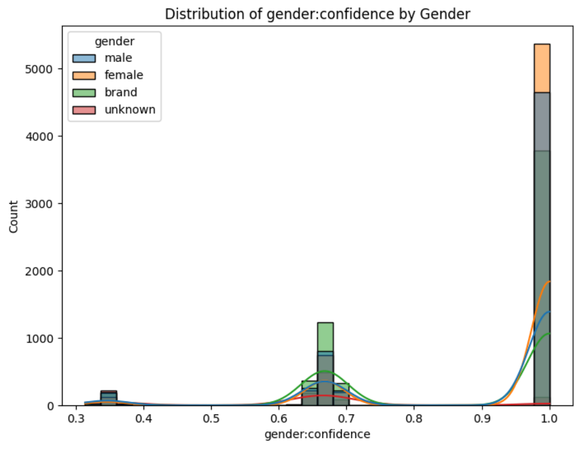


Figure 4: The distribution gender:confidence by gender

The key features are:

- `gender`: Categorical feature representing the user's gender or whether they are a brand.
- `tweet_count`: The number of tweets posted by the user.
- `retweet_count`: The number of times the user's tweets have been retweeted.
- `description`: A textual description provided by the user.
- `fav_number`: The number of likes received by the user.

2. Missing Data Analysis

Handling missing data was an important step in cleaning the dataset. We identified columns with significant missing values and decided on the following actions:

- **Dropping columns**: The columns `gender_gold`, `profile_yn_gold`, and `tweet_coord` were dropped because over 90% of the data was missing, rendering them uninformative for our analysis.
- **Filling missing values**: Missing values in the `description`, `tweet_location`, and `user_timezone` columns were filled with the placeholder `'Unknown'`. This approach allowed us to retain rows without losing information.
- **Removing rows with missing gender**: Rows with missing values in the `gender` column were removed since they represented a small percentage of the data, ensuring that the dataset remained focused on the human/non-human classification task.

The table below shows the missing data distribution before and after cleaning:

Column	**Missing Before (%)**	**Missing After (%)**
`gender_gold`	90%	N/A (dropped)
`description`	30%	0% (filled)
`gender`	2%	0% (rows removed)

3. Distribution of Key Features

Categorical Features

The `gender` feature was a primary focus of this analysis. A bar plot was used to visualize the distribution of gender categories in the dataset.

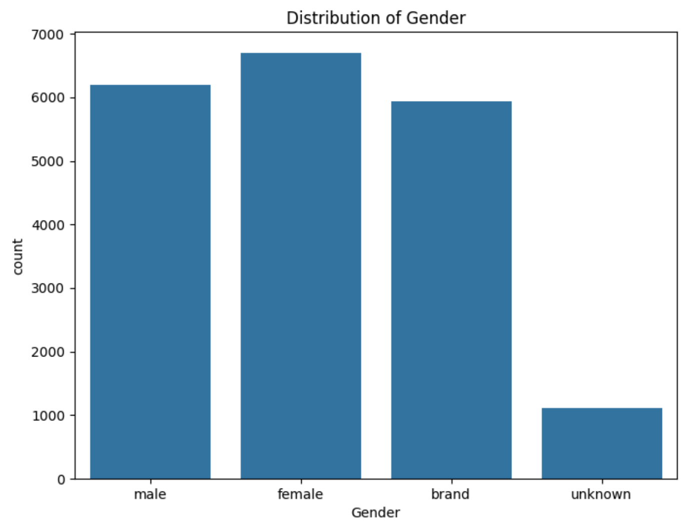


Figure 5: The distribution of gender

- **Result**: The majority of users are classified as male, followed by female, while a smaller subset is classified as brand. This class imbalance happens when performing classification of human/non-human, meaning that we may require delicate consideration during model training to avoid bias towards the majority class.

Numerical Features

The distributions of key numerical features like `tweet_count` and `retweet_count` were explored using histograms with a kernel density estimate (KDE).

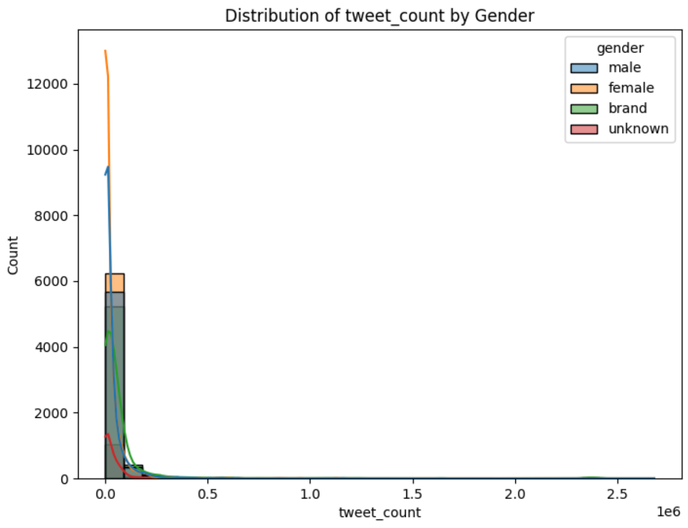


Figure 6A: The distribution of tweet_count by gender

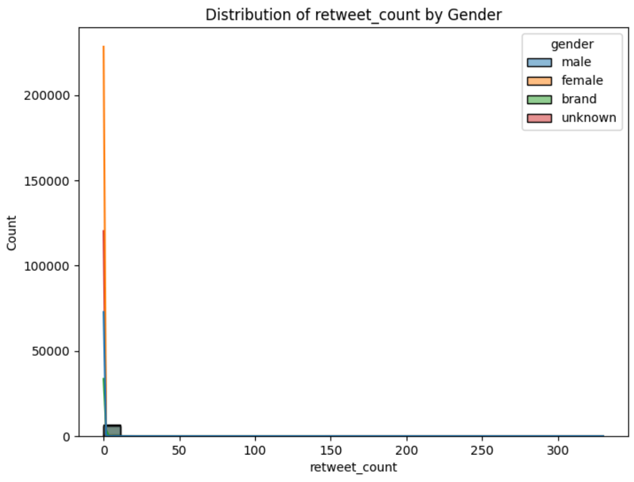


Figure 6B: The distribution retweet_count by gender

- **Tweet Count Distribution**: Most users have a relatively low number of tweets, with the majority tweeting fewer than 500 times. There are a few outliers with much higher tweet counts, which might represent highly active accounts or automated bots.

- **Retweet Count Distribution**: The retweet count shows a similar pattern, with most users receiving relatively few retweets. A small number of users have very high retweet counts, likely representing highly influential accounts.
- **Feature engineering:**
    - `account_age` to `profile_created_year`: from calculating the difference between the current date and the date when the Twitter profile was created. This provides insight into how long a profile has been active, which is a useful feature for identifying behavior patterns over time (e.g., new users vs. long-term users).
    - `tweet_count`, `retweet_count`, and `fav_number` to `tweet_per_day`, `retweet_per_day`, and `favorite_per_day`: From the original features, calculating per-day metrics by dividing these counts by the `account_age`. These features normalize the activity data, providing a standardized view of how frequently users engage on Twitter, regardless of their account age. This normalization helps prevent biases in clustering by ensuring that older accounts with more total activity aren't automatically treated as more important.

4. Correlation Analysis

A **correlation matrix** is a common technique to identify multicollinearity between features, ensuring that redundant features are removed before clustering. This step prevents overfitting and ensures that clustering is not biased by highly correlated features (James et al., 2013). Therefore the first correlation matrix in the figure was created to explore the relationships between numerical features such as `tweet_count`, `retweet_count`, and `fav_number`. The correlation heatmap is displayed below in figures 6 and 7:

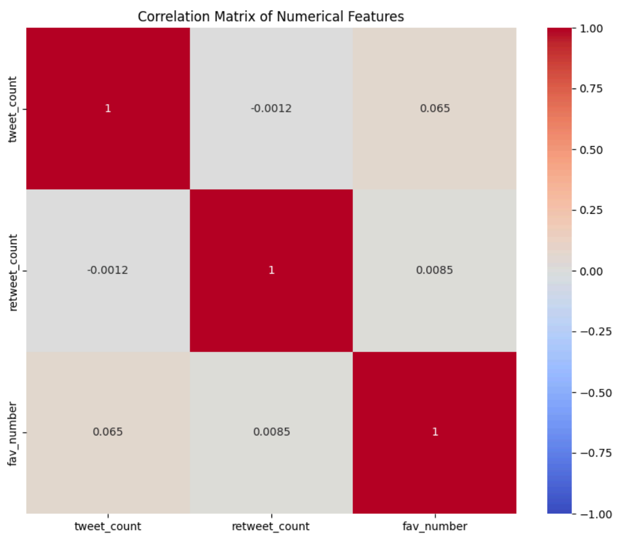


Figure 7A: The correlation matrix of numerical features

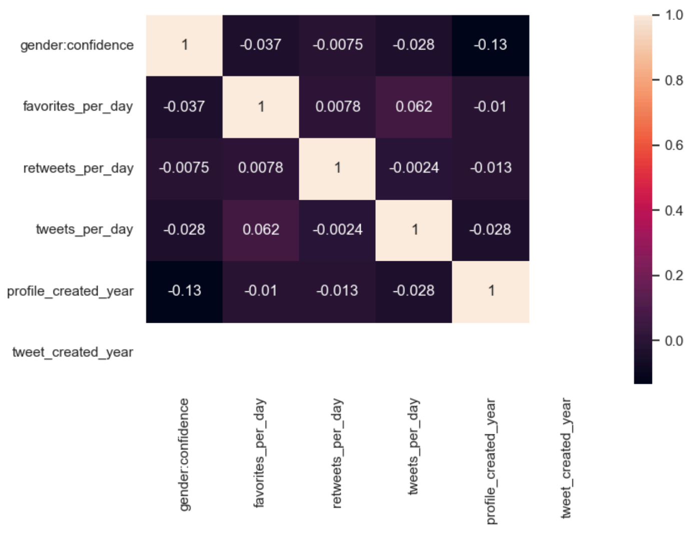


Figure 7B: The correlation matrix of numerical features after feature engineering

- **Key Findings before feature engineering**:

  - A moderate positive correlation (0.5~0.5 0.5) was observed between `tweet_count` and `retweet_count`, indicating that users who tweet more often tend to get retweeted more.
  - `fav_number` shows a weak correlation with both `tweet_count` and `retweet_count`, suggesting that the number of likes a user receives may not be strongly tied to their tweeting activity.
- **Key Findings after feature engineering**:
  - A slightly higher positive correlation was observed between `tweet_per_day` and `favorite_per_day`, indicating that users who tweet more often tend to show favor (favorite) more.
  - `gender:confidence` shows a weak correlation with all features, suggesting that the confidence in predicting gender might not be dependent or related with any numerical features.

5. EDA Visualizations

Using distribution plots segmented by **gender** and color-coded plots of link_color and sidebar_color aligns with the principles of exploratory data analysis (Tukey, 1977). Visualizing how different genders engage with the platform helps identify patterns that can be used to inform the clustering model (Han et al., 2011). Correspondingly, all the key visualizations (bar charts, histograms, scatter plots, and heatmaps) are included to support the insights drawn from the data.

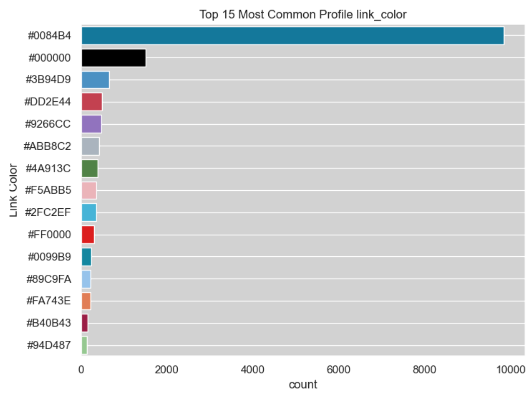


Figure 8A: Top 15 most used link colors

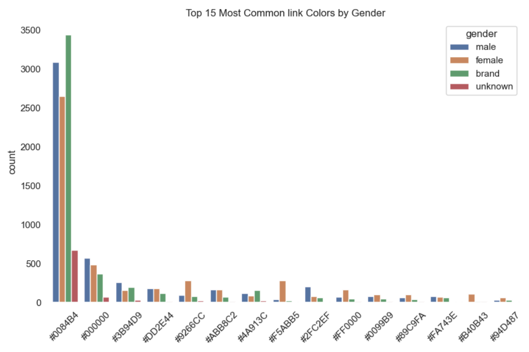
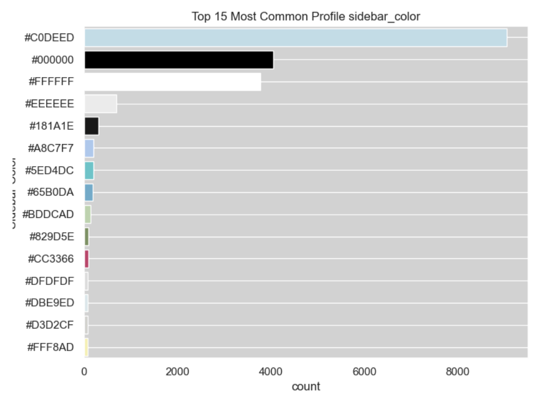


Figure 8B: Top 15 most used link colors by gender



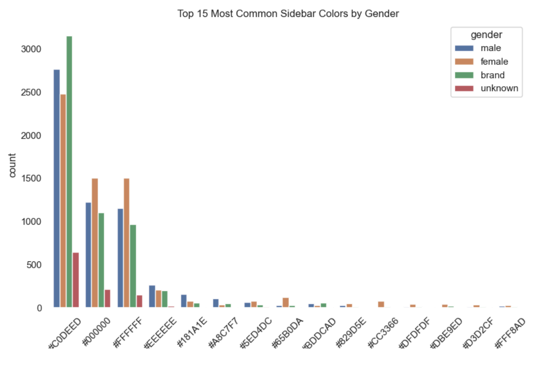


Figure 9A: Top 15 most used sidebar colors

Figure 9B: Top 15 most used sidebar colors by gender

According to all the plots in figure 8A, 8B, 9A, and 9B, it seemed that there were no explicit significant differences in using either sidebar and link colors by each gender. The pattern in choosing colors among all genders were indistinctive, yet there may be some unique color that could be used in a particular gender which can be investigated further.

Conclusion of EDA

The EDA revealed several important patterns in the data. Key insights such as class imbalance, outliers in numerical features, and correlations between user activity variables will guide the subsequent data preprocessing and modeling steps. Specifically, handling the imbalance between human and non-human accounts, scaling numerical features, and accounting for outliers will be important for building an accurate classification model.

Data Preprocessing

Data preprocessing is a critical step in preparing the dataset for machine learning. In this phase, we handled missing data, tokenized textual features, scaled numerical data, and transformed categorical variables for modeling. The following outlines the specific preprocessing steps taken:

1. Handling Missing Data

Several columns in the dataset contained missing values, and we handled them in the following ways:

- **Dropping columns with excessive missing values**: Dropping features with over 90% missing values is a standard practice to ensure that the remaining data is both informative and not overly influenced by missingness (Little &

Rubin, 2002).The columns gender_gold, profile_yn_gold, and tweet_coord had more than 90% missing values. Since they provided little useful information, we removed them from the dataset to reduce noise.

- **Filling missing values**: We filled missing values in the description, user_timezone, and tweet_location columns with a placeholder 'Unknown'. This strategy allowed us to retain important rows while marking the missing data clearly for potential downstream use.
- **Dropping rows with missing gender**: Since the gender column is crucial for the classification task, we dropped rows where the gender value was missing. These rows constituted a small portion of the dataset, so their removal did not significantly impact the data.

2. Extracting and Tokenizing Text Data

Text cleaning through tokenization, lemmatization, and lowercasing is essential in **natural language processing** to standardize text and remove noise (Manning et al., 2008). **TF-IDF Vectorization** transforms text into a numerical representation that emphasizes important terms while downplaying common words, making it suitable for clustering (Ramos, 2003). One of the key features in the dataset is the description column as well as text column, which contains user-provided textual data. We performed the following steps to process this column:

- **Text extraction**: We extracted the description and text columns from the dataset, recognizing that user-generated text may hold valuable information for the classification task (e.g., distinguishing between personal and brand accounts).
- **Tokenization**: The text data was tokenized, splitting each user's description into individual words or tokens. Tokenization is a necessary step for preparing text data for machine learning, as it converts free-form text into a structured format that can be analyzed or used as features in a model.
- **Combining tokenized descriptions with other meaningful features**: After tokenization, the resulting tokens were combined with other relevant features from the dataset (e.g., tweet_count, retweet_count). This allowed us to include the textual data as part of the overall feature set for classification.

3. Scaling Numerical Features

To ensure that the numerical features in the dataset were on a consistent scale, we applied **standardization** using StandardScaler. This method transforms numerical features to have a mean of 0 and a standard deviation of 1, which is critical for models that rely on gradient-based optimization or distance calculations.

- The features scaled include:

    - `tweet_count`: The number of tweets a user has posted.
    - `retweet_count`: The number of times a user's tweets have been retweeted.
    - `fav_number`: The number of likes a user has received.

By scaling these features, we ensured that large ranges in the data would not disproportionately affect the model's learning process.

4. Encoding Categorical Variables

Encoding **gender** as a numerical variable and frequency encoding **location** and **timezone** is a widely accepted technique in machine learning for categorical data. It allows models to use these variables quantitatively without inflating dimensionality (Pedregosa et al., 2011). Frequency encoding is particularly effective when the categorical features have a high cardinality, as it prevents overfitting (Zheng & Casari, 2018).

For the `gender` column, which contains categorical values, we converted the text labels into numerical codes to make them usable in machine learning models:

- **Gender Encoding**:
    - `male` → `0`
    - `female` → `1`
    - `brand` → `2`

Converting hex color values from `link_color` and `sidebar_color` to **RGB format** allows for color features to be treated as numerical data, which can be used quantitatively in the clustering model. This approach is supported by work in image and aesthetics-related clustering tasks (Zhang et al., 2020).

This encoding allows the model to treat `gender` as a binary variable, for further analysis in classification and regression, new encoding identifies `0` to represent human accounts and `1` represents non-human (brand) accounts. Note that both `male` and `female` were mapped to `0` to create a single "human" class for classification.

5. Final Data Cleanup

- **Removing irrelevant categories**: We filtered out rows where `gender` was classified as `unknown` since this did not provide meaningful information for our classification task.

- **Final structure check**: After completing the preprocessing steps, we verified the structure of the cleaned dataset and displayed the first few rows to ensure that the transformations were applied correctly.

Summary of Preprocessing Steps:

1. **Missing data**: Dropped unnecessary columns with excessive missing data and filled missing values with placeholders where appropriate.
2. **Text processing**: Tokenized the `description` column to handle textual data effectively.
3. **Scaling**: Standardized numerical features to ensure consistency in their ranges.
4. **Categorical encoding**: Encoded the `gender` variable to convert it into numerical labels for the classification task.
5. **Data cleanup**: Removed rows with irrelevant or missing values, leaving a cleaned and well-structured dataset for model training.

Conclusion of Preprocessing

The preprocessing steps were essential for preparing the dataset for model building. By handling missing data, scaling numerical features, and encoding categorical variables, we ensured that the dataset is clean, balanced, and in a format suitable for machine learning algorithms. Additionally, the tokenization of textual data allows us to incorporate descriptive information from the `description` column into the model, potentially improving the accuracy of our human/non-human classification task.

Clustering Algorithms

Approach to the Pre-processed Dataset

Given the complex nature of the dataset, which includes numerical, categorical, and text features, dimensionality reduction was crucial to handle the high dimensionality, especially after preprocessing the text data. UMAP (Uniform Manifold Approximation and Projection) was chosen for dimensionality reduction and visualization. UMAP is particularly effective for preserving both local and global structures, making it useful for visualizing high-dimensional data in lower dimensions while maintaining meaningful relationships between clusters (McInnes, Healy, & Melville, 2018). UMAP allowed us to project the dataset into 3D, potentially providing a clearer visualization of the clustering results (Becht et al., 2019).

Model Justification and Rationale

KMeans was chosen for its efficiency and ability to partition data into a predefined number of clusters. It works by minimizing the variance within clusters and is particularly useful when the dataset has clear groupings (Han et al., 2011). In the context of this dataset, KMeans was expected to separate Twitter profiles based on predefined features. The algorithm's computational efficiency, especially for larger datasets, makes it a suitable choice for this assignment, where the dataset involves numerous features and records.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise), on the other hand, was selected for its ability to identify clusters of arbitrary shape and detect outliers (Ester et al., 1996). Unlike KMeans, DBSCAN does not require a predefined number of clusters and can automatically adjust based on the data's density. This makes DBSCAN particularly well-suited for detecting misclassified or outlier profiles, such as bots or unusual user behaviors that might deviate from typical human-like interactions on Twitter. Additionally, DBSCAN's robustness in handling noisy data provides an advantage when dealing with messy, real-world social media datasets, where noise and outliers are often prevalent.

Metric Justification

Silhouette Score: This metric calculates how similar each data point is to its own cluster compared to other clusters. Higher scores (closer to 1) indicate better separation between clusters (Rousseeuw, 1987).
Calinski-Harabasz Index: Also known as the Variance Ratio Criterion, it measures the ratio of between-cluster dispersion to within-cluster dispersion, with higher values indicating better-defined clusters (Calinski & Harabasz, 1974).

These metrics were selected because they provide a clear understanding of the compactness and separation of clusters, especially when dealing with both dense and noisy data, as is the case in social media profiles.

Hyperparameter Tuning

Optuna is an open-source hyperparameter optimization framework known for its flexibility and efficiency in tuning machine learning models. It leverages advanced techniques like

Bayesian optimization and Tree-structured Parzen Estimators (TPE) to automate the hyperparameter search. Clustering algorithms like KMeans, DBSCAN, and HDBSCAN are highly sensitive to parameters such as `n_clusters`, `eps`, and `min_samples`, and manually tuning these can be time-consuming. Optuna automates this process by focusing on promising regions of the hyperparameter space, making it especially useful for improving clustering performance metrics like the Silhouette Score. It also supports multi-objective optimization, allowing the fine-tuning of multiple internal metrics. Moreover, Optuna efficiently explores the parameter space, automatically selecting the best combination based on an objective function (like Silhouette Score or Calinski-Harabasz Index) which deciding parameter choice is critical because poor parameters can lead to trivial results, like either all data points being in one cluster or each data point being its own cluster. In all experiments, every model was tuned to maximize their Silhouette score in order to gain the best capture of comparable profiles in each cluster.

Experiments

Regarding the data life cycle and the behaviors of the dataset after investigation in EDA, clustering algorithms were created to deepen further understanding of the underlying patterns in this dataset. 3 experiments were strategically conducted to determine how each set of features could affect the performance of 2 distinctive clustering models: KMeans and DBSCAN, measured by the selected metrics. All models in 3 experiments were tuned by Optuna to identify the best parameter and achieve computational efficiency. All gender features were dropped to prevent their undesirable influence and to prove if clustering models would be able to categorize the remaining profiles based on gender.

The experiment 1 was conducted with the dataset that consisted of all feature's types, followed by the experiment 2 that aimed to evaluate how the model performed when only numerical and categorical features were used without dimensional reduction techniques applied on, excluding the text features. The last experiment was for evaluation of the model's performance using only text-based features processed through TF-IDF. K-Means and DBSCAN were implemented on the same feature set to see how well they identify clusters of Twitter profiles based on their activity, profile information, and text data. Both algorithms will use TF-IDF for text features.

All in all, 3 experiments meant to compare clustering models performance and potentially, discover unrevealed patterns in the dataset that leads to identifying misclassification in human/non-human profiles.

Experiment 1: K-Means vs. DBSCAN with All Features Using TF-IDF

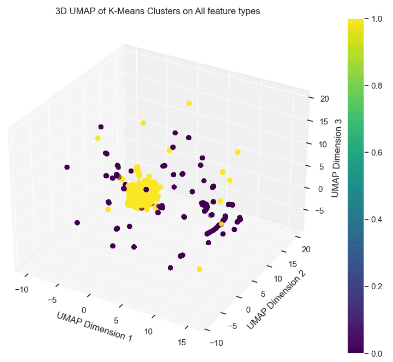


Figure 10: KMeans with UMAP in experiment 1

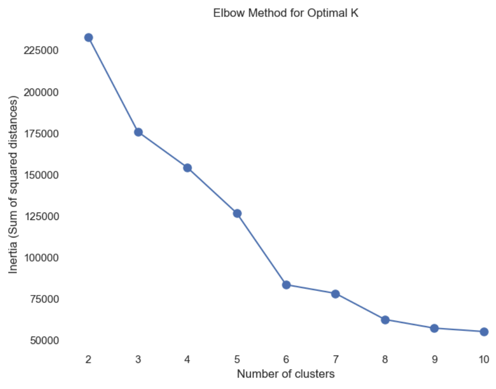


Figure 11: Elbow method of KMeans in experiment 1

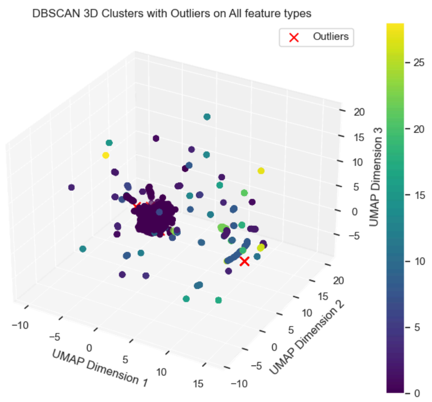


Figure 12A: DBSCAN with UMAP in experiment 1 (diagonal view)

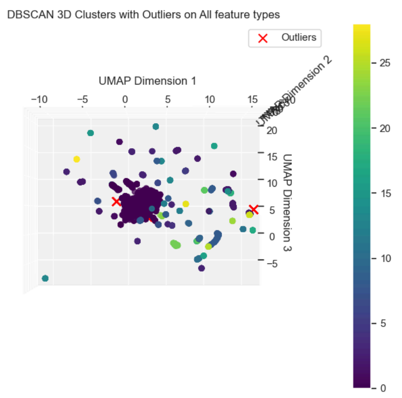


Figure 12B: DBSCAN with UMAP in experiment 1 (top view)

Results and Discussion from Experiment 1

Model	Silhouette score	Calinski-Harabasz Index
KMeans	**0.7922**	**30,077.87**
DBSCAN	**0.4712**	**2,554.79**

From figure 10, the optimal solution for KMeans was achieved with two clusters (n_clusters=2). This produced a high Silhouette score of 0.7922 and a Calinski-Harabasz Index of 30,077.87. The cluster analysis indicated a balanced distribution of gender labels within Cluster 0, while Cluster 1 exhibited more concentrated profiles with all 3 genders. However, one limitation of K-Means is its sensitivity to noise and outliers. Profiles that did not conform well to a specific cluster may have been assigned to the nearest centroid, which could forcibly classify abnormal profiles into some clusters regardless of gender. This drawback could be significant when attempting to detect misclassified profiles in noisy data (profiles that are borderline between human and non-human). K-Means also requires specifying the number of clusters in advance, which can introduce bias if an incorrect number is chosen. Corresponding to figure 11 which suggested the optimal number of clusters in range of 3 to 6.

Using an eps value of 1.906 and min_samples=20, DBSCAN produced a Silhouette score of 0.4712 and a Calinski-Harabasz Index of 2,554.79. DBSCAN's results shown in figure 12A and 12B revealed fewer distinct clusters, with almost 100 points classified as noise, highlighting its ability in identifying outliers but its challenge in creating well-defined clusters compared to KMeans as it detected above 40 clusters in this dataset.

KMeans outperformed DBSCAN in terms of both cluster separation and cluster density, as evidenced by the significantly higher Silhouette Score and Calinski-Harabasz Index. This suggests that KMeans is more effective when dealing with the complete feature set, including text, due to its efficiency in partitioning the data into compact, well-separated clusters. DBSCAN's noise detection, while useful, resulted in fewer identifiable clusters (Ester et al., 1996). For this dataset, DBSCAN might outperform K-Means in detecting mislabeled non-human profiles because it can identify profiles that lie outside dense human clusters.

Experiment 2: K-Means vs. DBSCAN with Numerical and Categorical Features (No dimensional reduction before clustering)

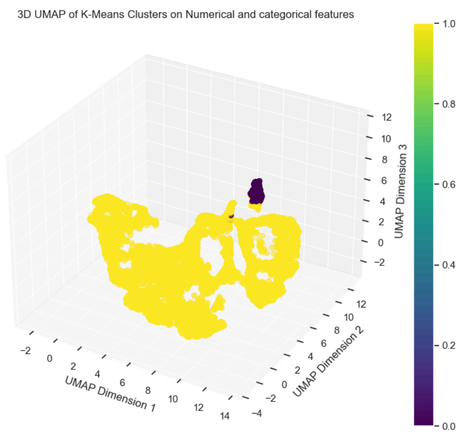


Figure 13A: KMeans with UMAP in experiment 2 (diagonal view)

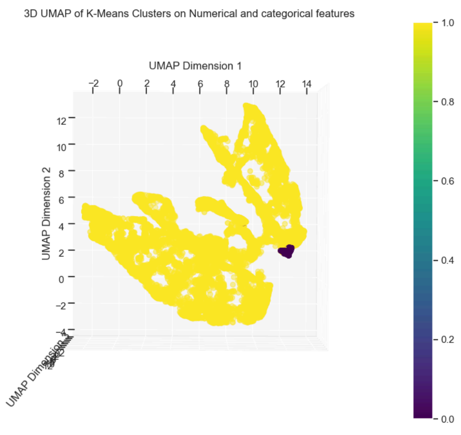


Figure 13B: KMeans with UMAP in experiment 2 (top view)

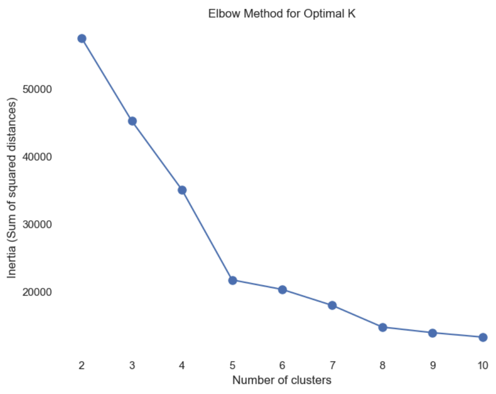


Figure 14: Elbow method of KMeans in experiment 2

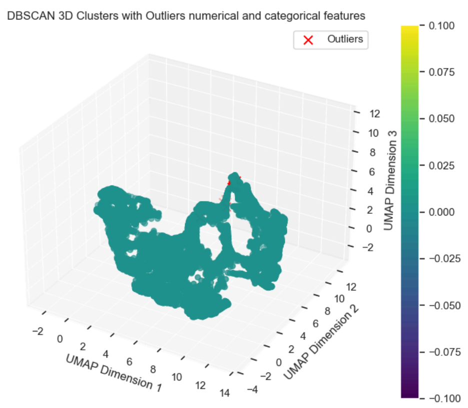


Figure 15: DBSCAN with UMAP in experiment 2

Results and Discussion from Experiment 2

Model	Silhouette score	Calinski-Harabasz Index
KMeans	**0.7076**	**4,482.75**
DBSCAN	**0.7695**	**182.57**

KMeans created two clusters, achieving a Silhouette score of 0.7076 and a Calinski-Harabasz Index of 4,482.75. Cluster 0 was significantly smaller than Cluster 1 in terms of gender distribution, indicating that KMeans barely could detect differences in numerical and categorical attributes between profiles. Another interesting point is that the Elbow Method in figure 14 found the different number of clusters (ranging from 4 to 5) from Optuna, implicitly telling us that 2 clusters might not discover well-defined patterns of the dataset. From figure 13A and 13B, KMeans highlights 2 distinct clusters represented by different colors. The central yellow cluster indicates a highly dense region of data points, which may represent a large group of users sharing similar features or behaviors. The smaller purple cluster suggests profiles that are notably different from those in the yellow cluster. Since K-Means assumes that the data is evenly distributed and attempts to create spherical clusters, it may struggle with the complex structure of data (as evident from the irregular spread in the UMAP dimensions). Nonetheless, K-Means possibly can help group users that exhibit similar patterns, which may indicate potential misclassifications or anomalies when compared to their expected groupings.

With an eps of 1.903 and min_samples=3, DBSCAN yielded a Silhouette score of 0.7695 and a Calinski-Harabasz Index of 182.57. The number of detected clusters remained low according to figure 15, with several points classified as noise, particularly those with gender=1 (female). Additionally, in contrast to KMeans, shows a denser region of data points (in teal) with outliers. DBSCAN repeatedly excels at identifying clusters of arbitrary shapes and densities without requiring the specification of the number of clusters beforehand, unlike

K-Means. The red outliers are users that do not fit well into any of the detected clusters, which might indicate profiles that deviate significantly from typical user behaviors (either human or non-human). These outliers could be misclassified profiles that warrant further investigation, suggesting that DBSCAN is particularly useful for identifying anomalous profiles in the dataset.

In this experiment, DBSCAN achieved a higher Silhouette score than KMeans, indicating that DBSCAN better handled the numerical and categorical features, particularly when it came to separating dense clusters. However, the Calinski-Harabasz Index was lower, reflecting DBSCAN's tendency to classify many data points as noise. KMeans still provided compact clusters, even with lower cluster separation than in Experiment 1.

Experiment 3: K-Means vs. DBSCAN with Text Features

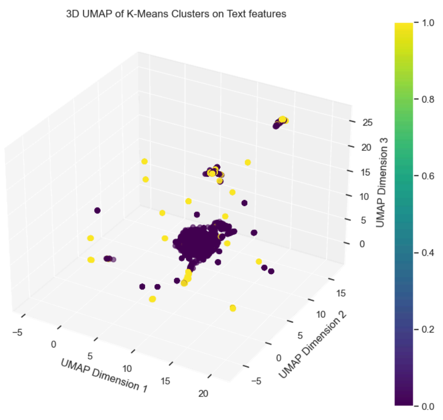


Figure 16A: KMeans with UMAP in experiment 3 (diagonal view)

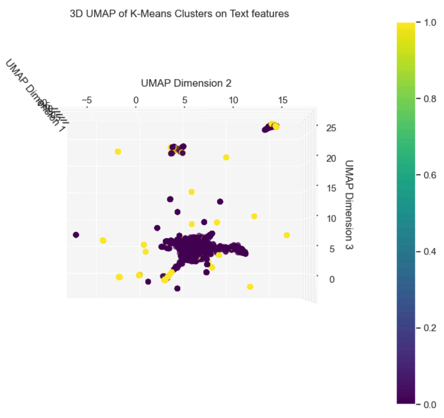


Figure 16B: KMeans with UMAP in experiment 3 (side view)

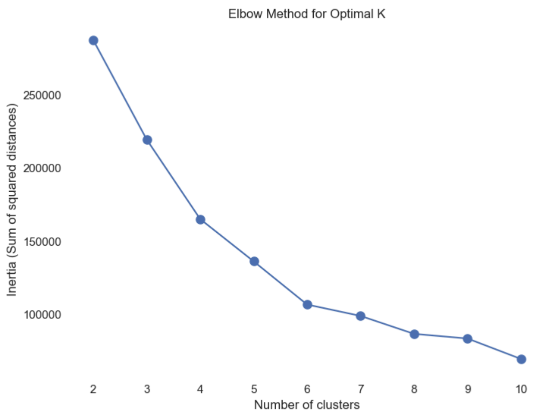
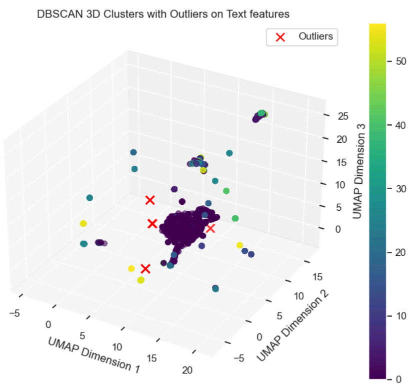


Figure 17: Elbow method of KMeans in experiment 3



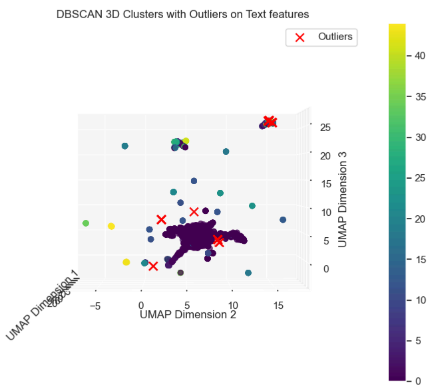


Figure 18A: DBSCAN with UMAP in experiment 3 (diagonal view)

Figure 18B: DBSCAN with UMAP in experiment 3 (side view)

Results from Experiment 3

Model	Silhouette score	Calinski-Harabasz Index
KMeans	**0.7513**	**10,709.34**
DBSCAN	**0.5791**	**1500.54**

With two clusters as can be seen in figure 16A and 16B, KMeans produced a Silhouette score of 0.7513 and a Calinski-Harabasz Index of 10,709.34. The text features allowed KMeans to effectively separate profiles based on distinct clusters, with the cluster compositions reflecting different gender distributions. The cluster analysis indicated a comparable balanced distribution of gender labels within Cluster 1, while Cluster 0 exhibited a numerous number of profiles yet also a balanced distribution of gender labels. However, the visualization indicates the imbalance between 2 clusters. The profiles that fall in cluster 0 excessively outnumbered the profiles in cluster 1. In addition, the suggested number of clusters provided by the Elbow method in figure 17 was in the range of 4 to 6, apparently in contrast with the best parameters provided by Optuna. This could suggest that 2 clusters may not be suitable and could demonstrate the interpretable or meaningful clusters regarding the objective of this project.

Meanwhile, with An eps value of 1.726 and min_samples=12 were used for DBSCAN, which achieved a comparable Silhouette score of 0.5791 and a Calinski-Harabasz Index of 1,500.54. However, in figure 18A and 18B, DBSCAN again, as in figure 18A and 18B, seemed to struggle with sparse, high-dimensional text data, as evidenced by a large number of located clusters. Positively, The dense areas suggest the algorithm has grouped profiles with similar writing styles, keyword frequency, or engagement in specific topics. The smooth transitions between some points and the spread of the clusters show that DBSCAN clusters profiles without assuming a specific shape, making it useful for discovering organic groupings in natural language-based features. Potentially, DBSCAN could help segment Twitter profiles effectively by distinguishing dense clusters and highlighting outliers. These outliers, likely representing non-human profiles, can be important for identifying misclassified entities in the dataset.

Overall, even though KMeans could demonstrate better cluster compactness and separation when applied to text data, as reflected by the high Calinski-Harabasz Index and DBSCAN struggled to handle high-dimensional text features, leading to a relatively lower Silhouette Score and a high noise rate, indicating that density-based clustering algorithms like DBSCAN may not be ideal for text clustering tasks (Ester et al., 1996), DBSCAN can be a useful approach in identify misclassified profiles.

Discussion on Clustering Implementation Across 3 Experiments

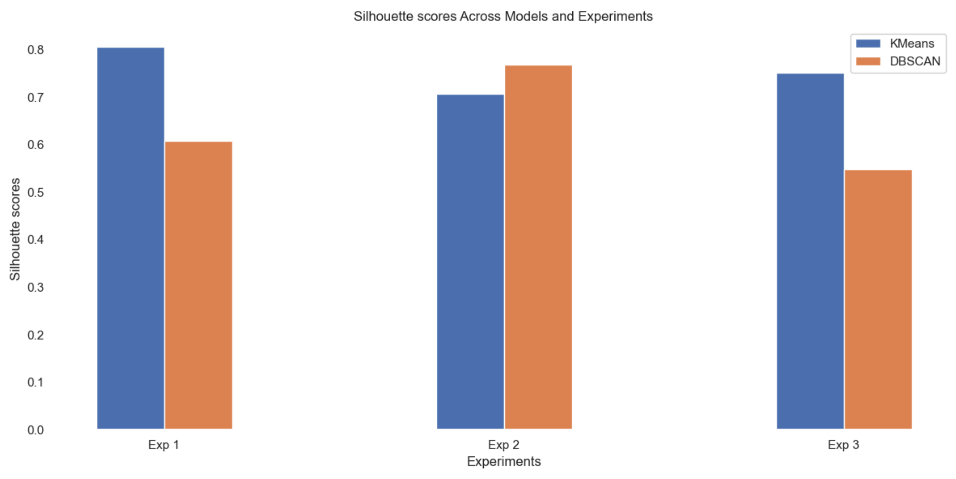


Figure 19: Silhouette scores across all experiments

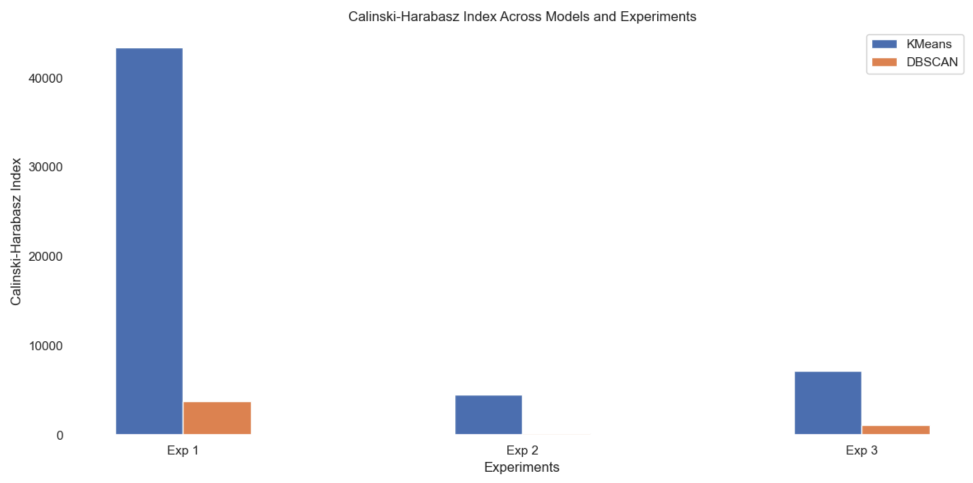


Figure 20: Calinski-Harabasz Index across all experiments

With observation from figure 19 and 20, KMeans consistently outperformed DBSCAN across almost all experiments in terms of cluster compactness and separation evaluated by available metrics. In particular, KMeans handled the diverse feature sets well, especially when combining numerical, categorical, and text features. The high Calinski-Harabasz Index and Silhouette Scores for KMeans demonstrate that it could be well-suited for this dataset, particularly in cases where distinct clusters exist.

Even though KMeans successfully clustered user profiles based on available features, it did not explicitly differentiate between human and non-human profiles beyond what the existing features could capture. While it performs well for separating structured data into clear clusters, it is less effective in identifying outliers or profiles that do not fit neatly into the defined groups. Since KMeans forces every data point into a cluster, profiles that may not belong to a typical user group such as bots or misclassified profiles are likely to be clustered within the nearest group. This limitation suggests that misclassified profiles (e.g., bots labeled as humans) may blend into regular clusters, and their detection becomes challenging without further analysis.

Noteworthily, DBSCAN was able to detect profiles that deviated from typical behavior patterns, classifying them as noise points. These noise points could represent misclassified profiles, such as bots, that do not fit into any of the well-defined clusters. For instance, if a profile has extremely high tweet activity but low engagement (retweets, favorites), DBSCAN is more likely to classify such a profile as an outlier. By doing so, DBSCAN effectively identifies potential non-human users or users whose behavior deviates from the norm, offering a robust mechanism for detecting misclassified profiles. Nevertheless, `gender:confidence` of outliers in all 3 experiments ranged from 0.6 to 1 which required more investigation of this feature accuracy or its importance in regression and classification.

Furthermore, All dominant clusters detected by KMeans and DBSCAN always represent a well balanced proportion of 3 genders: male, female, brands, possibly indicating that human and non-human profiles cannot be categorized by all existing features. As such, future work can address additional pre-processing steps like reducing the dimensionality further with other techniques or employing more advanced feature selection techniques could improve the model's performance in identifying meaningful clusters. Also, further tuning of the `eps` and `min_samples` parameters could enhance the model's ability to detect more distinct groups while maintaining a balance between cluster cohesion and outlier detection. More hyperparameter tuning techniques can be implemented to compare how models perform differently and make possibly more concise justification with more aspects than heavily relying on Optuna. Another work that can be done in the future is to investigate the `profile` feature (images) with deep learning models to further capture the outstanding characteristics between human and non-human profiles.

Regression Algorithms

The regression part of the task focuses on exploring the feature “Gender confidence” in the dataset. Assuming that this feature is entirely accurate in the dataset is, in itself, an assumption. Therefore, the regression part focuses on exploring this assumption and checking whether 2 different regression models will come to the same results.

The general idea here is to attempt to train a regression model on a randomized subset of the dataset with the “Gender confidence” feature as the target regression value. The subset was chosen to be approximately 60% of the total data. Subset size is chosen to avoid overfitting. The hope is that the model will be able to pick up on a pattern different to that which the people creating the dataset used to decide gender confidence. Datapoints were for example the dataset has a gender confidence of say 90%, but the regression model predicts a gender confidence of say 50% should be marked as a datapoint where there is an increased chance of the user being misclassified as human / non-human.

Keep in mind that a classification method with gender confidence as an input feature is likely more suited to the task, as identifying mistaken profiles is directly linked to a classification task. However, performing regression might give valuable insight to the gender confidence feature, which is likely the most important feature to identify mistakenly classified profiles, and can give additional insight to the results from the classification model.

Model Justification

2 regression models were explored for the task. The first model was a standard gradient boosted regression model from the . The reason this model was chosen was due to its ability to capture complicated, nonlinear relationships inherent in data, without being too susceptible to overfitting to the dataset (even when a large amount of features are present, which is the case in this dataset). This ability allows it to approximate nonlinear functions in the feature space meaning that it can pick up on even complex patterns (James, Witten, Hastie, & Tibshirani, 2021). As there is a high probability that the pattern that decides gender confidence might be quite complex this should allow the model to pick up on a genuine pattern that decides gender confidence. Also, its ability to highlight feature importance provides valuable insights into how gender confidence is affected by the other features in the dataset.

The other model that was explored was a simple linear regression. This is somewhat of a naïve model as it will only detect linear patterns in the dataset (James, Witten, Hastie, & Tibshirani, 2021). However, this simplicity might give more interpretable results and could be very suitable if it is able to achieve a high accuracy of the dataset, indicating that the actual pattern deciding the gender confidence could indeed be linear. As the dataset has a very large feature space (especially when text processing methods such as vectorizing the text into a numerical feature vector is used) there is a somewhat large chance for the model to overfit to the data it is trained on. To combat this a regularization could be performed. Also keep in mind that the model will likely be too simple to pick up on the pattern deciding the gender confidence, as it is likely more complex than a linear relationship in the feature space.

Experiments

The models were to be tested in two different experiments: the first with text preprocessing features from the description and text columns, and the second excluding them.

In all experiments users were marked as potentially “mistaken” based on a simple thresholding. If the predicted gender confidence is below 0.85 and (in addition) is lower than 0.1 of the original dataset gender confidence, it indicates spots where the dataset has a markedly higher confidence than the predicted confidence. This indicates records with the potential to be incorrectly identified as human / non-human and are marked as “mistaken” by the models.

Gradient Boosted Regression Tree

Wit Text Features

As seen in the figure below, the gradient boosted regression tree achieves a relatively low MSE in both the training and test data. This is a positive result as it indicates that the model does not heavily overfit to the training data, meaning the model has the ability to generalize and is therefore more likely to have discovered a genuine pattern in the data that can be used to infer the gender confidence. This is also reflected in the fact that not only data points in the train data were marked as potentially mistaken. However, the train MSE is lower then the test MSE and more of the potentially mistaken data points are not in the train data. This indicates that there is some overfitting present in the model.

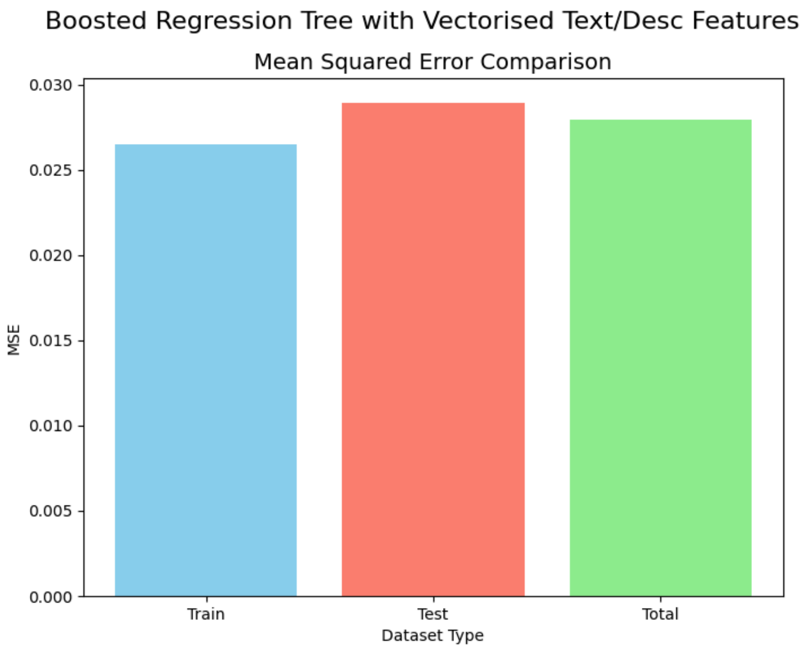


Figure 21: Plot of Mean Square Error for boosted regression tree

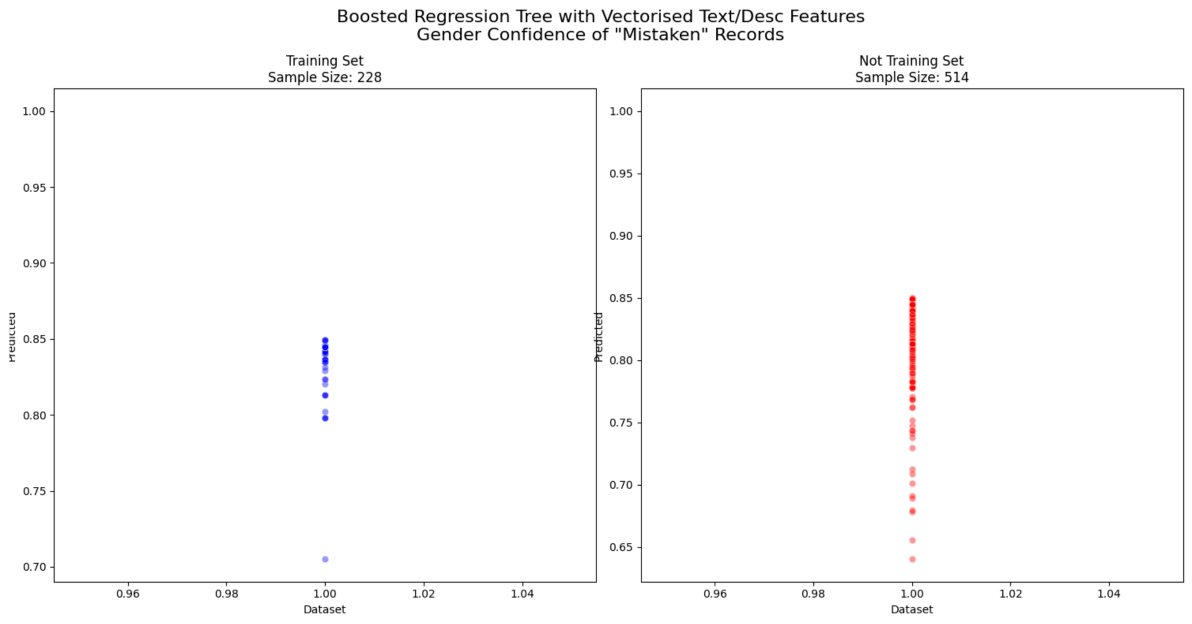


Figure 22: Mistaken records boosted regression tree

Based on the feature importance the vectorized features play an important role in deciding the gender confidence, when they are available. Furthermore, the distribution of the predicted gender confidence versus the gender confidence in the dataset shows that the boosted regression tree in general predicts a higher gender confidence than that assigned in the dataset. In addition, it is only on data points where there is a gender confidence of 1.0 in the original dataset that have been marked as potentially mistaken as human / non-human.

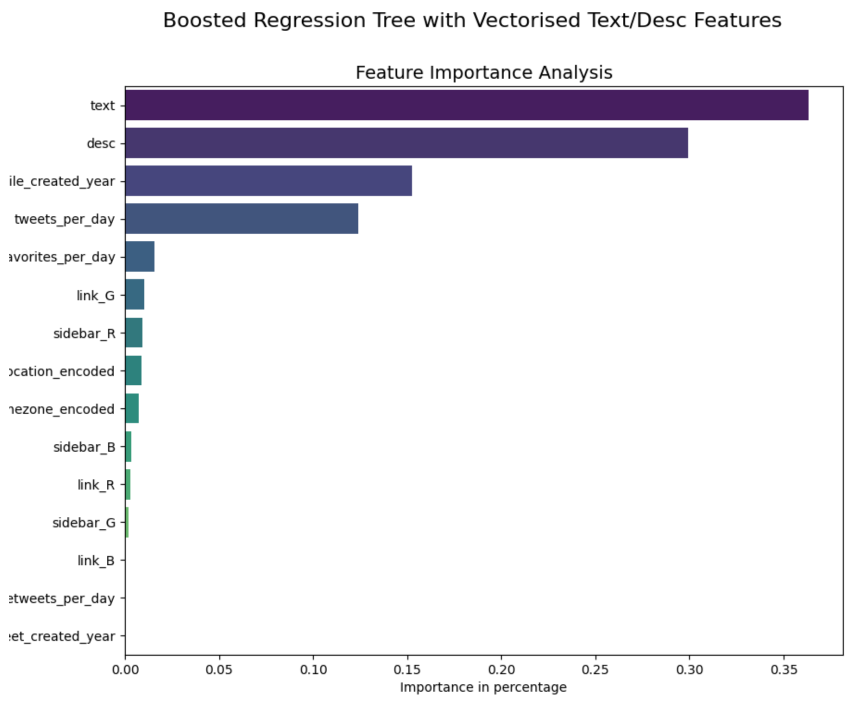


Figure 23: Plot of feature importance for boosted regression tree

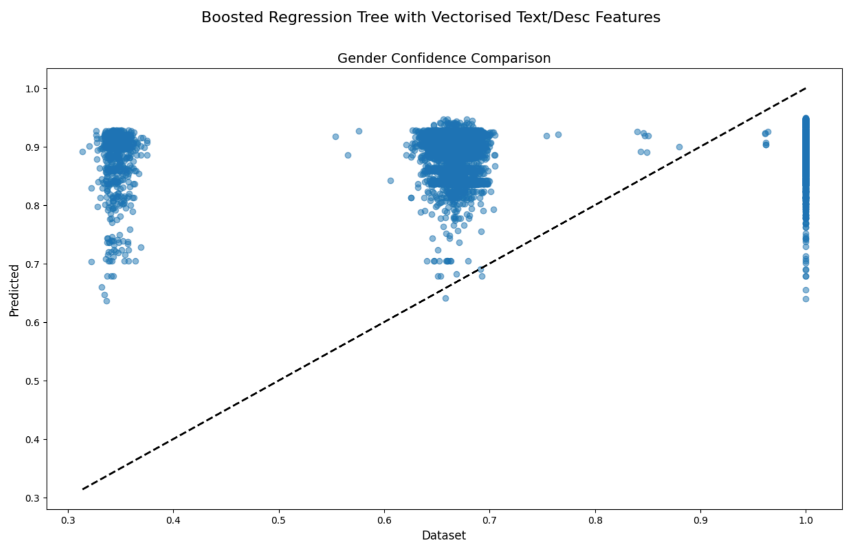


Figure 24: Distribution of predicted gender confidence versus dataset gender confidence for boosted regression tree

Without Text Features

Again, the MSE in both train and test is relatively low, however it is slightly higher then when including the text features. This again indicates that the model does not heavily overfit to the training data. Also, the training MSE is slightly higher (at about 0.02 higher) than before, while the test MSE did not increase by much (only 0.002) indicating the model is overfitting slightly less now. This is a very natural result as the feature space is vastly reduced. This is also reflected in that the ratio between potentially mistaken records in the training data compared to not in the training data is slightly lower then with the text features. The ration with text features is: 0.444 compared to 0.477 without them.

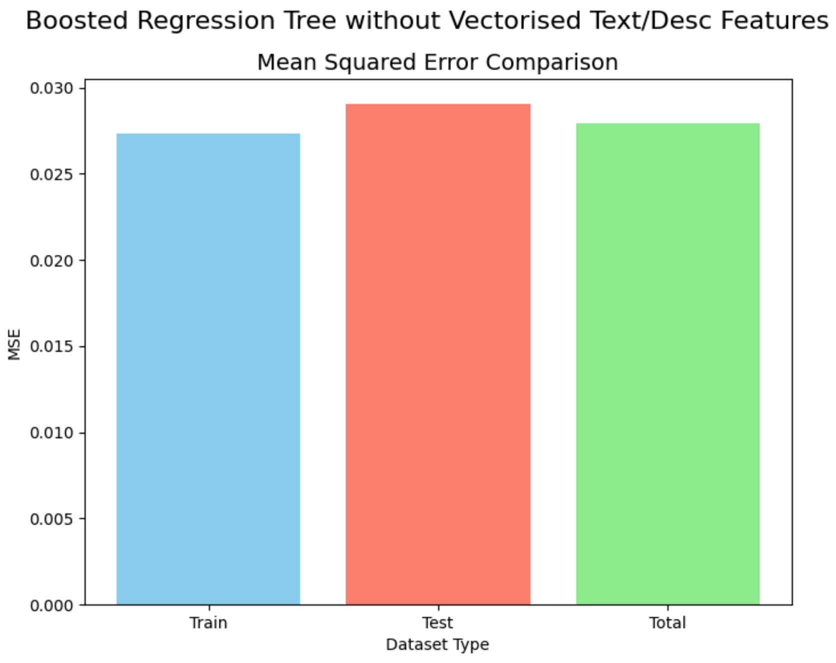


Figure 25: Plot of Mean Square Error for boosted regression tree without text-based features

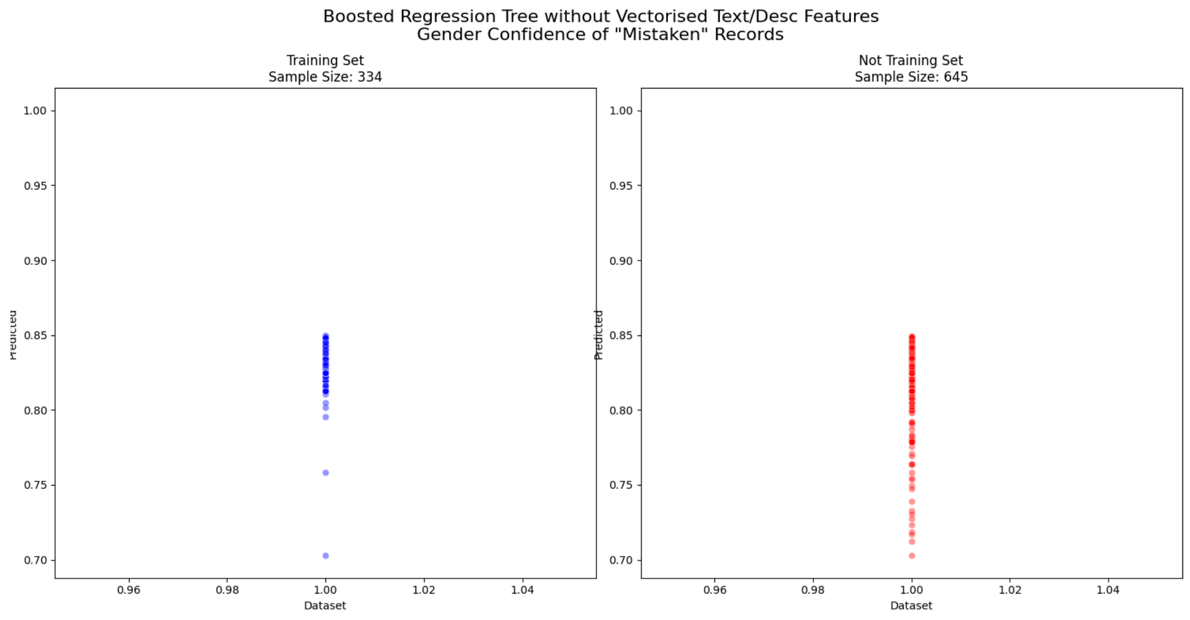


Figure 26: Mistaken records boosted regression tree without text-based features

Again, it is only on data points where there is a gender confidence of 1.0 that the regression model predicts potentially mistaken as human / non-human. Also, the model predicts higher gender confidence than the data set on average.

Based on the feature importance we see that “tweets_per_day”, “profile_created_year” and “favorites_per_day” are the most important features. This makes sense, as these were also the top features after the text-based features in the experiment that included text preprocessing.

Based on the feature importance we see that “tweets_per_day”, “profile_created_year” and “favorites_per_day” are the most important features. This makes sense, as these were also the top features after the text-based features in the experiment that included text preprocessing.

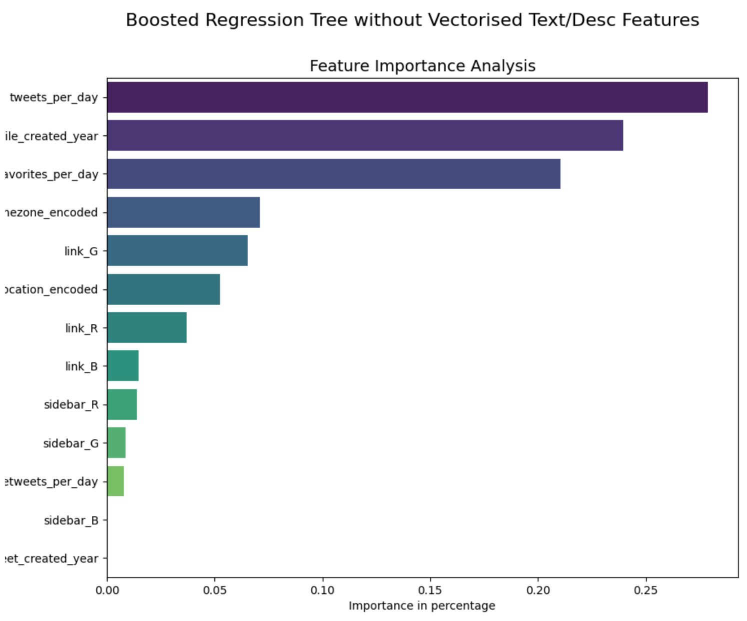


Figure 27: Feature importance for boosted regression tree without text-based features

Linear Regression

With Text-based Features

As seen in the figure below, the train MSE is significantly lower than the test MSE meaning that the model seems to overfit to the training data. The ratio between potentially mistaken records in the training data compared to not in the training data is also significantly higher than the gradient boosted regression tree being 0.286. This suggests that the model does not generalize well and may have failed to capture a genuine pattern in how to determine gender confidence.

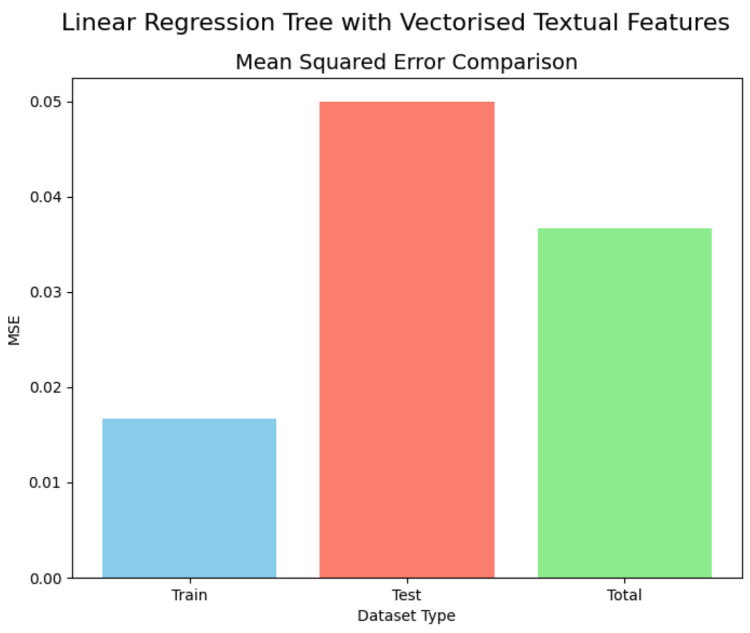


Figure 28A: Plot of Mean Square Error for linear regression with text-based features

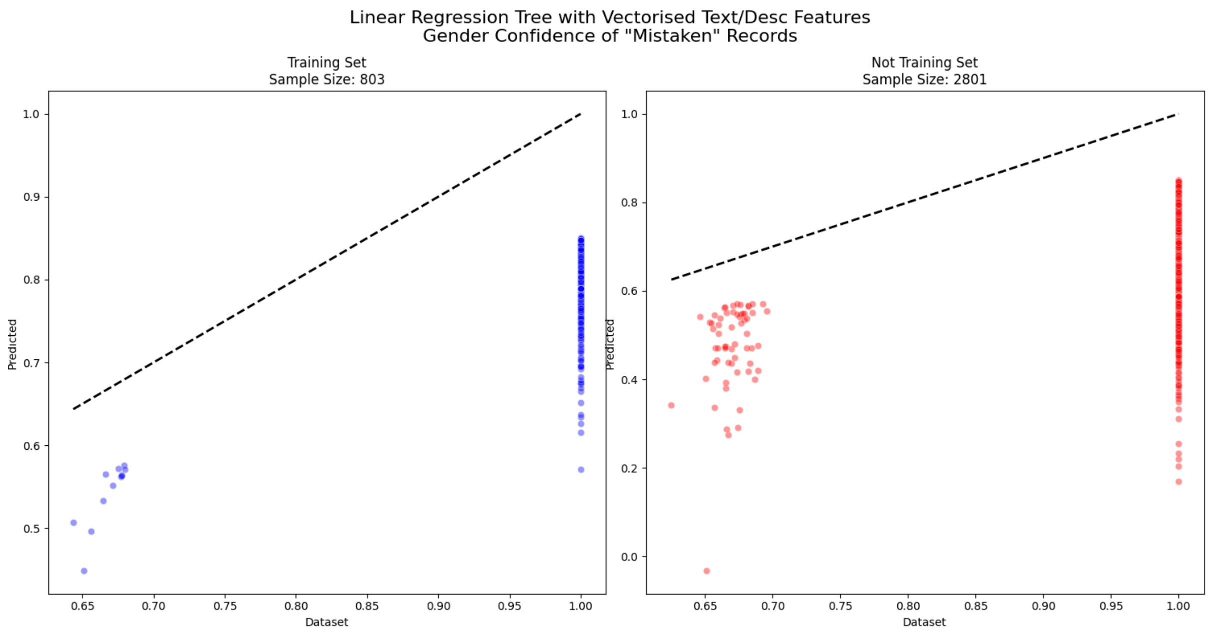


Figure 28B: Plot of mistaken records for linear regression with text-based features

However, the distribution of the predicted gender confidence versus the gender confidence is less extreme towards predicting a higher gender confidence than that assigned in the dataset than in the case of the gradient boosted regression tree. This means that the model identifies more profiles as potentially mistakenly classified as human / non-human. Of especial interest, is that the model does not only mark data points with a gender confidence of 1 in the original dataset as potentially misclassified. This means that if one were to assume that a gender confidence of 1 can be trusted in the original dataset, by assuming that the team has verified these users genders through definitive means, the model will still give results that can be used to identify mistakenly classified users.

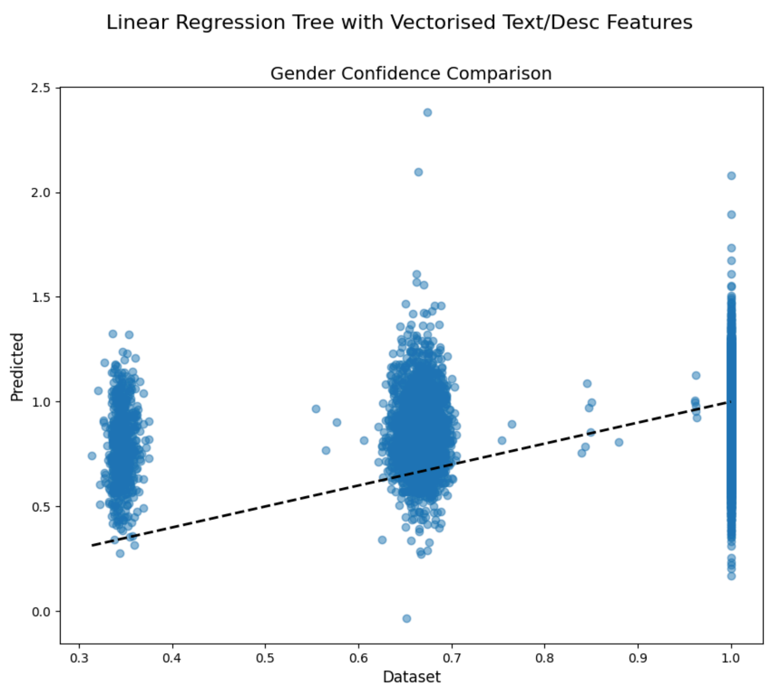


Figure 29: Distribution of predicted gender confidence versus dataset gender confidence linear regression with text-based features

Without Text-based Features

As seen in the plots below the model is overfitting much less than with the text features which, as before, is to be expected as the feature space is vastly reduced.

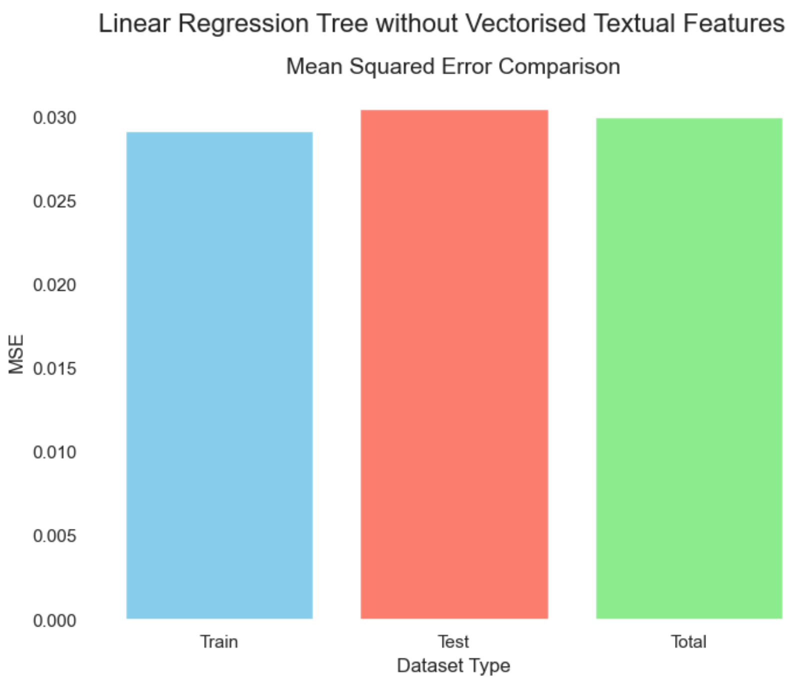


Figure 30: Plot of Mean Square Error for linear regression without text-based features

The distribution of the predicted gender confidence is also much closer to what was seen in the gradient boosted regression tree. Also worth noting is that there are much less users that are flagged as potentially mistaken with the linear regression without text-based features and the gradient boosted tree. Also the users that are detected as potentially mistaken all have a gender confidence of 1 in the original dataset.

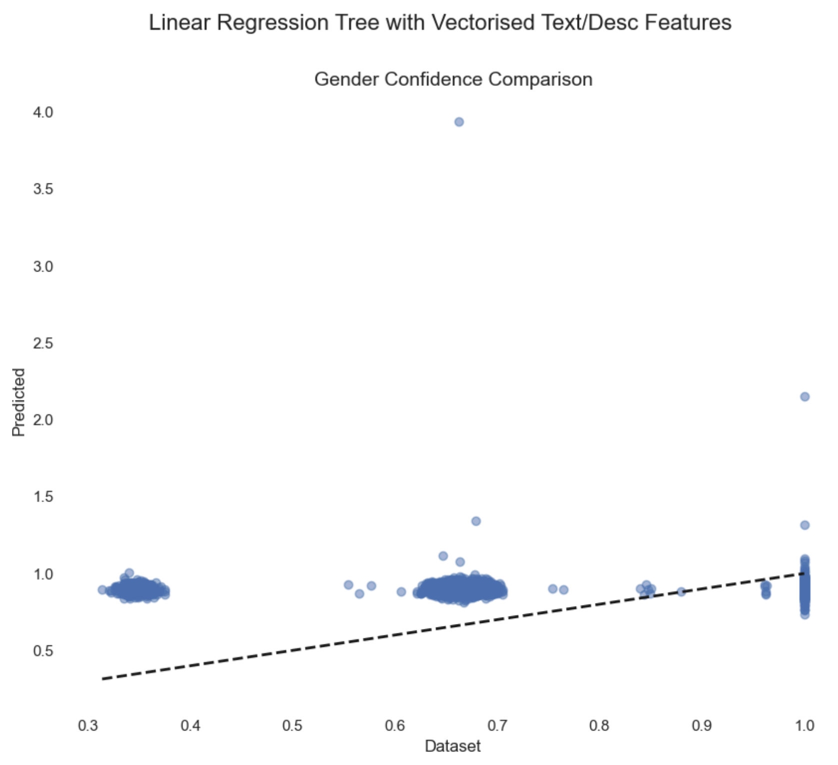


Figure 31: Distribution of predicted gender confidence versus dataset gender confidence for linear regression without text-based features

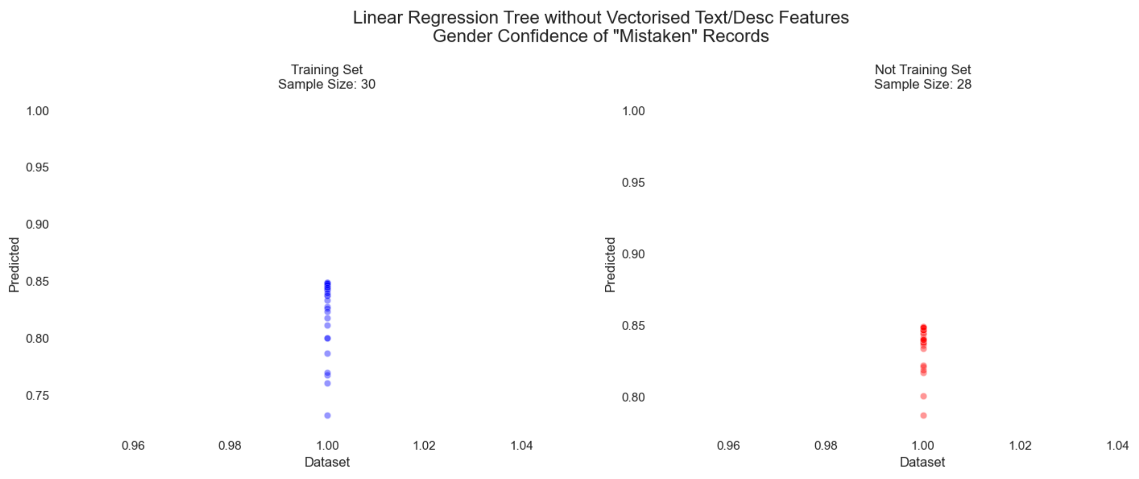


Figure 32: Mistaken records for for linear regression without text-based features

Recommendations

If one can assume that the users with a gender confidence of 1 can be trusted with absolute confidence in the original dataset then the users marked as potentially misclassified by the gradient boosted regression tree and the linear regression without the text-based features cannot be used. The only results that can be used in that case, is from the linear regression with the text-based features in the description / text column. However, the model in this case appears to be highly prone to overfitting, so the results should be used with caution.

If one were to assume that the users with a gender confidence of 1 in the dataset cannot be trusted with absolute confidence, then the results from the gradient boosted regression tree show promise in detecting falsely identified users in the twitter dataset.

Association Rules

FP-Growth Algorithm

The FP-Growth algorithm was applied to discover frequent patterns between the tokenized user descriptions and their corresponding gender in the dataset. This method assists in identifying correlations between the gender categories (0, 1, 2) and word usage in user descriptions, which may suggest profiles that are human or non-human.

Key Steps :

1. **Tokenization and Binary Encoding**:
   - Text input in the description field was tokenized into individual words, or tokens. Then, using multi-hot encoding, these tokens were converted into a binary format where each column indicates whether a certain token exists in a user's description.

2. **Combining Data**:
   - The gender column, which is pre-categorized as 0 (male), 1 (female), and 2 (other), has been combined with the binary-encoded token columns.

3. **FP-Growth Application**:
   - The **FP-Growth algorithm** was applied to the combined dataset with a minimum support threshold to extract frequent itemsets, representing tokens or gender values that frequently co-occur.
   - The lift metric, which gauges the degree of correlation between various tokens and gender values, was used to create **association rules** from these frequently occurring itemsets.

Findings from Frequent Patterns

Finding combinations of token and gender that commonly occur together is one of the analysis's main takeaways. Below is a representation of the top frequent itemsets, which highlights the most common token-gender pairs.

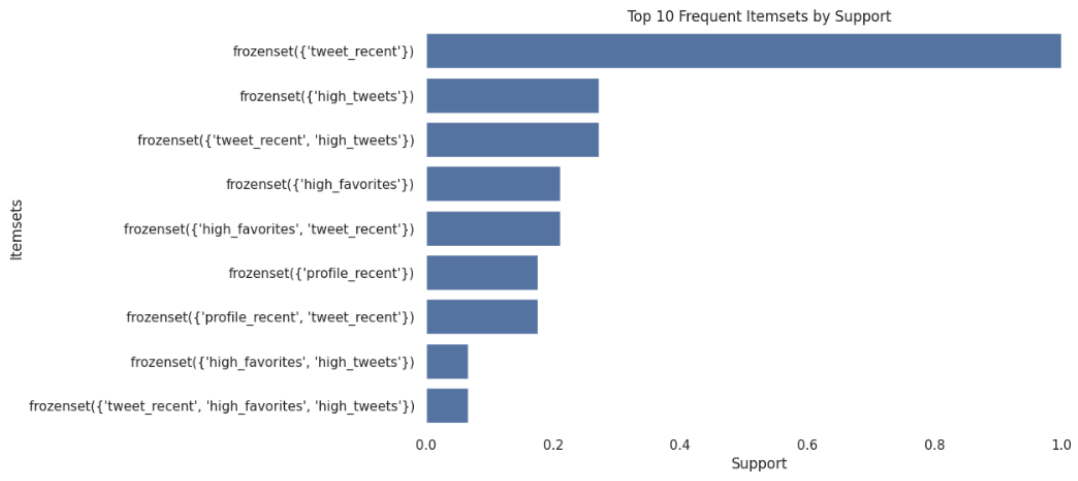


Figure 33: Frequent Itemsets by Support

Association Rules Scatter Plot: This scatter plot illustrates the strength of the generated association rules by plotting **confidence** against **lift**, helping to visualize the most meaningful and strong rules.

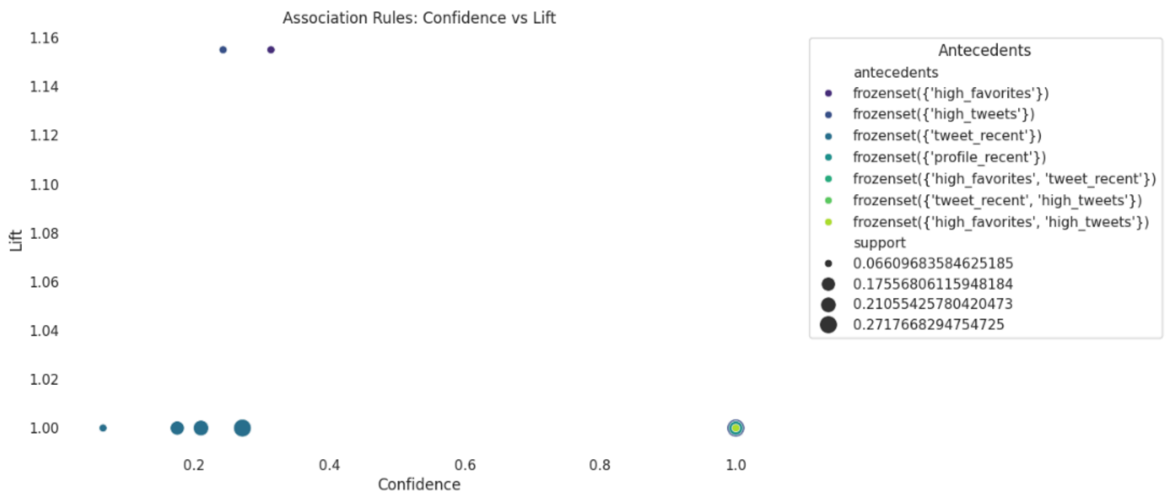


Figure 34: Association Rules: Confidence vs Lift

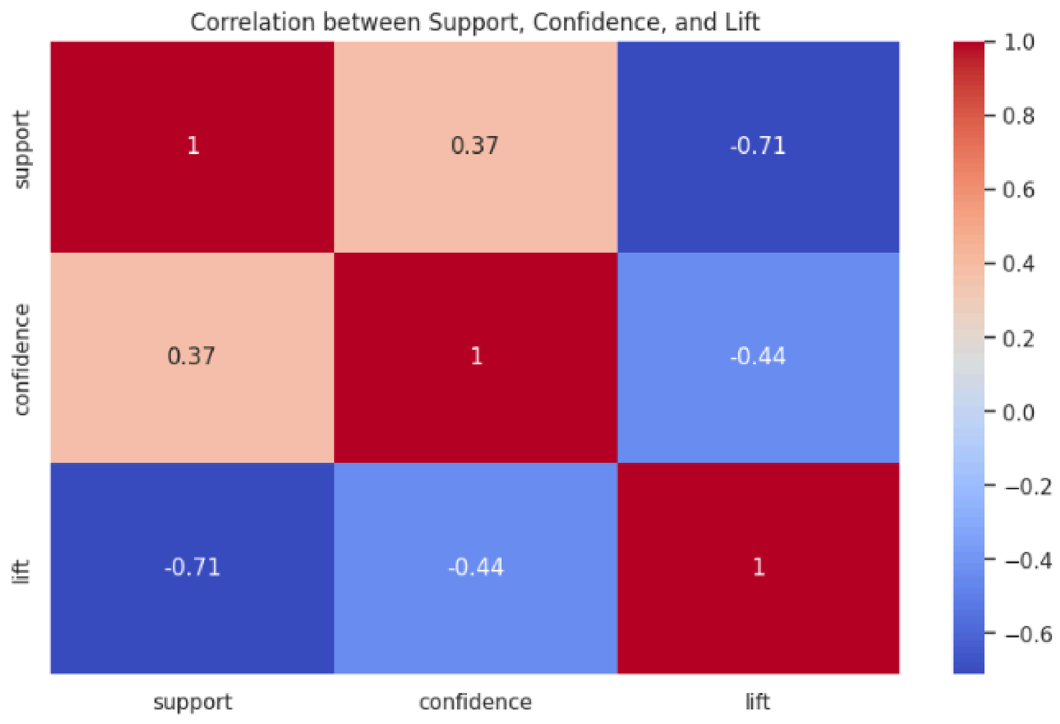


Figure 35: Correlation between Support, Confidence, and Lift

Classification Algorithms

In this analysis, we employed three machine learning models—Random Forest, XGBoost, and LightGBM—to classify social media profiles into three categories: male, female, and brand/bot. The primary objective was to identify non-human or bot accounts, which are often key sources of misinformation on social networks. By comparing these models' performance, we aimed to determine which classifier could most accurately distinguish between human and bot profiles. Each model was evaluated using metrics such as accuracy, precision, recall, and F1-score, with a particular focus on how well they handle the challenging task of differentiating between human profiles and automated accounts.

The selection of Random Forest, XGBoost, and LightGBM for this classification task was driven by their strengths in handling complex datasets and providing robust performance. Random Forest, being an ensemble method, is known for its ability to reduce overfitting by combining multiple decision trees and providing a stable performance across various datasets. XGBoost was chosen due to its efficiency and accuracy in classification tasks, as it utilizes gradient boosting, which helps in optimizing model performance by minimizing errors in each iteration. LightGBM has its superior speed and memory efficiency, particularly as its a large dataset as well as its effectiveness in managing sparse data and high-dimensional features. These models offer a balance between interpretability, accuracy, and computational efficiency, making them suitable for our goal of classifying profiles and detecting misinformation on social networks.

Experiment & Results

The experiment aimed to assess the performance of three classifiers—Random Forest, XGBoost, and LightGBM—by comparing their predicted values against the actual dataset labels. The objective was to identify potential misclassifications and evaluate how well each model could differentiate between the three classes: male (Class 0), female (Class 1), and brand or non-human profiles (Class 2). By doing so, the goal was to determine each classifier's ability to accurately predict social network profiles, especially in identifying non-human or bot accounts.

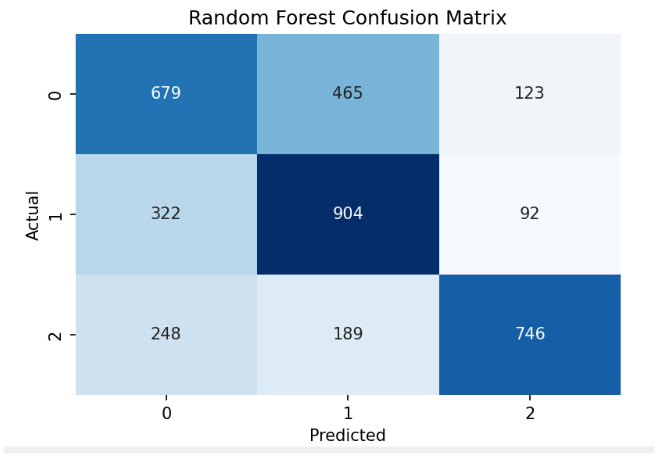


Figure 36

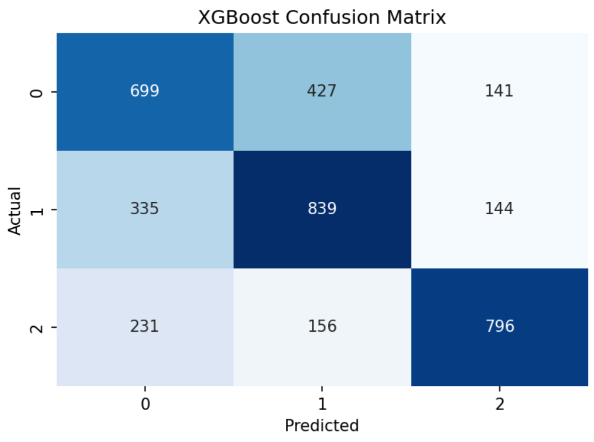


Figure 37

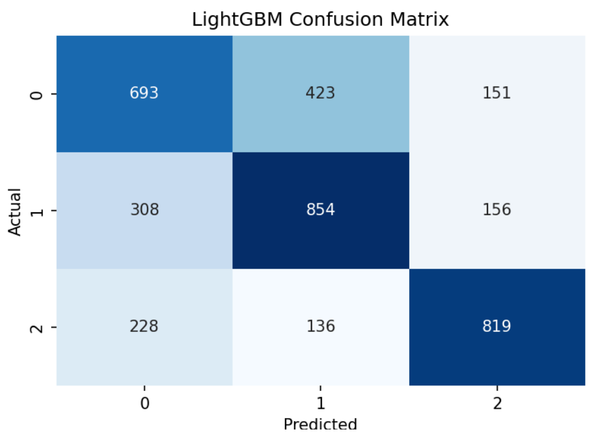


Figure 38

The classification results from Random Forest, XGBoost, and LightGBM models show varying performance levels across the three target classes: male (Class 0), female (Class 1), and brand/bot (Class 2). Overall, LightGBM performs the best, with an accuracy of 63%, followed by XGBoost at 61.99%, and Random Forest at 61.8%. Despite the similarities in accuracy, the models differ in how well they handle specific classes.

For Class 0 (Male), all models struggle to distinguish males accurately. Random Forest and XGBoost show nearly identical precision and recall (0.54–0.55), while LightGBM slightly improves with a precision of 0.57 and recall of 0.55. However, this class has the lowest overall performance across all models, indicating that male profiles are harder to separate from other classes, leading to frequent misclassification.

Class 1 (Female) performs better, particularly in recall, which means the models correctly identify a larger number of female profiles. Random Forest offers the highest recall at 0.68, but XGBoost and LightGBM are not far behind, with recall values around 0.64–0.65. Precision for this class remains moderate across the board, with the highest from LightGBM (0.61), showing that while more female profiles are detected, there are still some false positives.

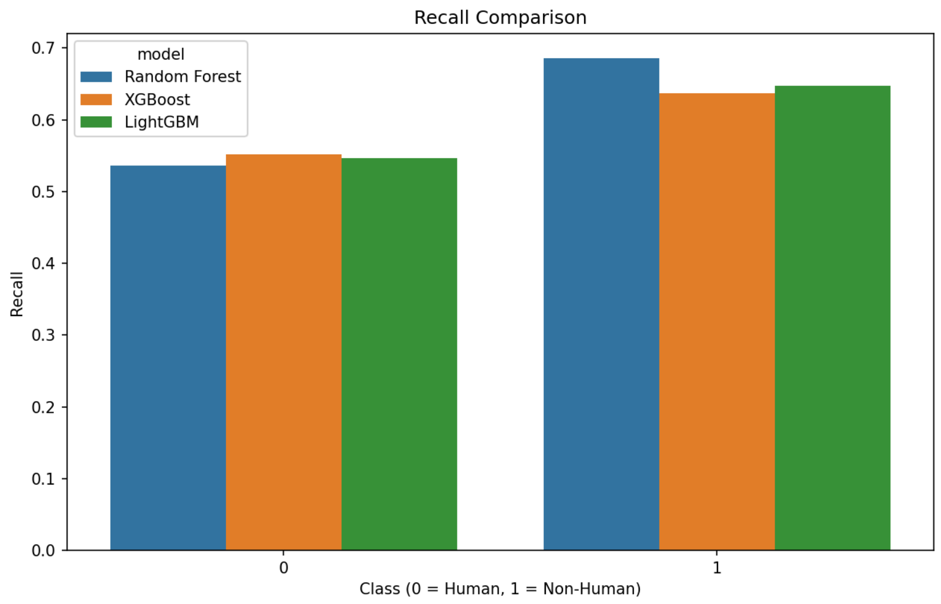


Figure 39

Class 2 (Brand/Bot) consistently performs the best across all models, with LightGBM showing the highest precision (0.73), recall (0.69), and F1-score (0.71). This indicates that all models are better at distinguishing bots from human profiles, with LightGBM offering the most balanced and reliable classification. The ability to identify bots accurately is crucial in the context of your project, as brands or bots are key sources of misinformation on social networks.

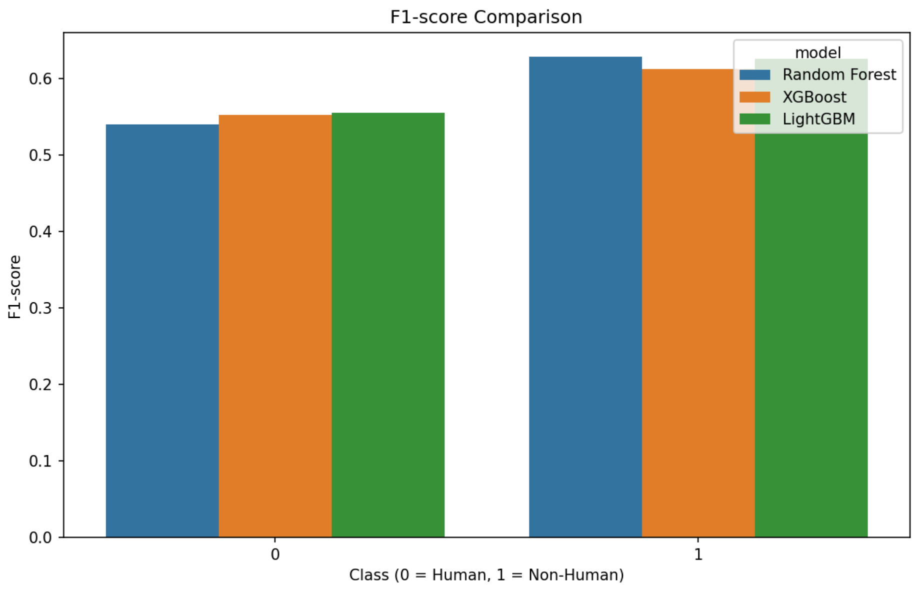


Figure 40

In summary, LightGBM stands out as the best-performing model, particularly in identifying bots, which aligns well with your goal of detecting misinformation spreaders (Class 2). However, the challenge lies in improving the classification of human profiles, especially males, where all models face difficulties. Using hyper- parameters and fine tuning for this to achieve a better result crashed the model as it took too much computational power.

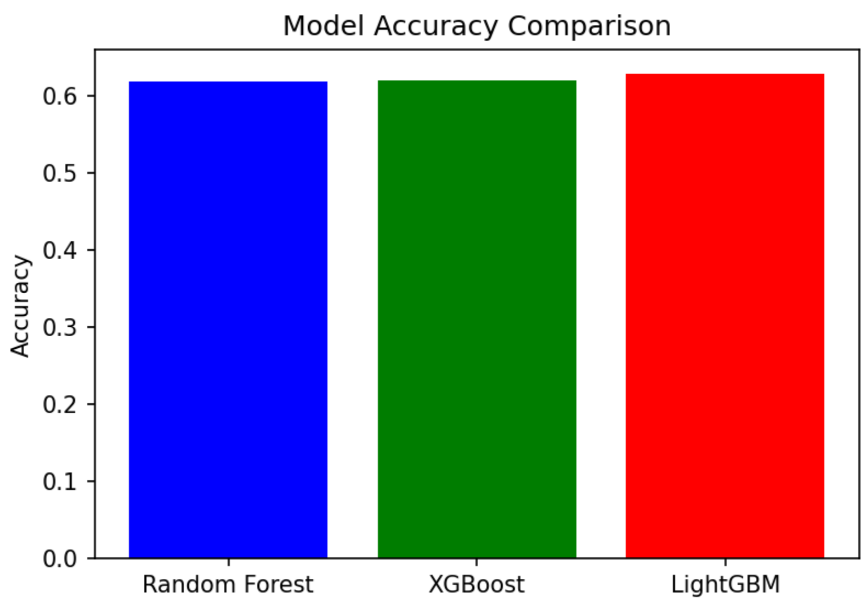


Figure 41

Thus, LightGBM was taken as used as a metric to check against any misclassification but since the model didn't have a high enough accuracy to concretely say which ones were misclassified and which ones were in-accuracies as its trained and tested on the same dataset.

Task 4

Task 4 aims to provide a few suggestions on the amendment process. This task relies mostly on the regression and the classification parts in the previous tasks. These parts are supposed to identify mistakenly recorded, as human or non-human, Twitter profiles. There are possibly different sets of such mistaken records that are captured by the separate tasks, and the amendment process attempts to consider a union of all captured sets.

It should be remembered that the original dataset may not be considered as the absolute truth in human-or-not-human classification. According to the dataset's publisher, CrowdFlower (CrowdFlower, 2016), the target features in question, `gender` and `gender:confidence`, are based on subjective judgment of project contributors. For this task, however, we consider the records with `gender:confidence` above a certain threshold as correct ones, known as the ground truth. In addition, future work may be able to address this issue by analyzing more data (e.g. verified status, followers to following ratio) from Twitter, or even manual validation, to establish the absolute truth (Imran et al., 2014).

Strategy

Two essential aspects are recommended for the amendment process: feature importance and similarity computation. The process starts with a set of all mistaken records captured by previous tasks. According to our regression experiments, the two most important features in identifying these records are textual (i.e. `text` and `description`), followed by two numerical features (i.e. `profile_created_year`, `tweet_count_per_day`).

After we perform exploratory analysis on the mistaken records in terms of these features, we compute similarity between each mistaken record and the ground truth. From the computed list, the human-or-not-human labels of the correct records with the best similarity scores, corresponding to each mistaken record, are used for amendment.

Methods

Three sub-tasks are included in the amendment process. The first sub-task is to analyze the mistaken records in terms of the four features. This will look at the distribution of values, investigate certain values (i.e. most frequent tokens in `text` and `description`, most frequent year of profile creation, and most frequent tweet frequency), and attempt to explain why the profiles with these specific values are mistakenly recorded.

The second sub-task is to compute similarity between every mistaken record and the ground truth. The similarity scores used for amendment are based on the Cosine similarity for vectorised textual features. Future work may use the computed similarity scores to compute confidence levels for the amendments.

The third sub-task is to conduct a comparative analysis of profiles before and after amendment. This task aims to determine whether the amended records have changed after the process and also analyze how they affect the overall dataset statistics. The output is a summary report of amendment results, which can be used for future manual checks.

Results

A very small portion, just 2.47 percent, of mistaken records are changed after the amendment process. This is because the process takes a union of all mistaken records captured by the previous tasks. A big portion of records in this input set are flagged as mistaken by one algorithm and as correct by others.

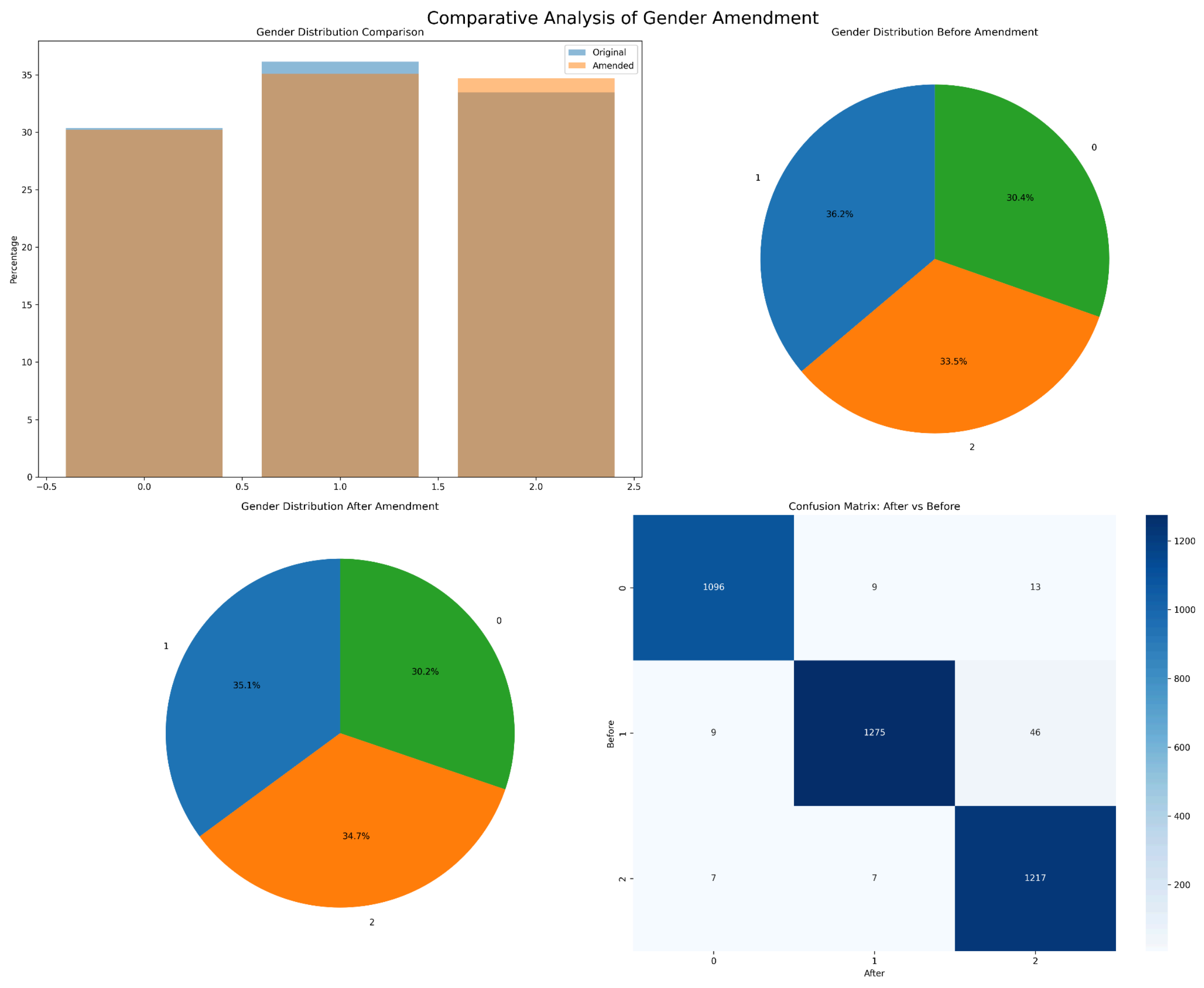


Figure 42: Amendment Summary Report

References

Han, J., Pei, J., & Kamber, M., *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2011.

James, G., Witten, D., Hastie, T., & Tibshirani, R., *An Introduction to Statistical Learning*, Springer, 2013.

Little, R. J., & Rubin, D. B., *Statistical Analysis with Missing Data*, John Wiley & Sons, 2002.

Manning, C., Raghavan, P., & Schütze, H., *Introduction to Information Retrieval*, Cambridge University Press, 2008.

Pedregosa, F., Varoquaux, G., Gramfort, A., et al., *Scikit-learn: Machine Learning in Python*, JMLR, 2011.

Ramos, J., *Using TF-IDF to Determine Word Relevance in Document Queries*, Proceedings of the First International Conference on Machine Learning, 2003.

Tukey, J. W., *Exploratory Data Analysis*, Addison-Wesley, 1977.

Van Buuren, S., *Flexible Imputation of Missing Data*, CRC Press, 2018.

Zhang, Y., Xu, X., & Zeng, J., *Color-based Aesthetic Feature Learning for Image Aesthetic Quality Assessment*, IEEE Transactions on Multimedia, 2020.

Zheng, A., & Casari, A., *Feature Engineering for Machine Learning*, O'Reilly Media, 2018.

Sandi M.P.K.U., *Unsupervised Clustering Algorithms: K-Means vs HAC vs DBSCAN*, Medium, 2020. Available at: https://medium.com/@sandi.mpku/unsupervised-clustering-algorithms-k-means-vs-hac-vs-dbscan-5947c9f5f2b9

Kaufman, L., & Rousseeuw, P. J., *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley, 1990.

Calinski, T., & Harabasz, J., *A dendrite method for cluster analysis*, Communications in Statistics, 1974, 3(1), pp. 1-27.

Jurafsky, D., & Martin, J. H., *Speech and Language Processing*, Prentice Hall, 2009.

Mikolov, T., Chen, K., Corrado, G., & Dean, J., *Efficient Estimation of Word Representations in Vector Space*, arXiv preprint arXiv:1301.3781, 2013.

Allaoui, M., Kherfi, M. L., & Cheriet, A., *Considerably Improving Clustering Algorithms Using UMAP Dimensionality Reduction Technique: A Comparative Study*, Image and Signal Processing, Springer, 2020.

Shalev-Shwartz, S., & Ben-David, S., *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, 2014.

Fortuna, P., & Nunes, S., *A Survey on Automatic Detection of Hate Speech in Text*, ACM Computing Surveys (CSUR), 2018, 51(4), pp. 1-30.

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M., *Optuna: A Next-generation Hyperparameter Optimization Framework*, Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019.

James, G., Witten, D., Hastie, T., & Tibshirani, R., *An Introduction to Statistical Learning: with Applications in R*, Springer, 2021.

CrowdFlower, *Twitter User Gender Classification*, Kaggle, n.d. Available at: https://www.kaggle.com/datasets/crowdflower/twitter-user-gender-classification

Imran M., Sajjad M., Imran M., *Twitter User Classes: A Classification Approach*, 2014. Available at: https://mimran.me/papers/moeen_imran_sajjad_twitter-user-classes_2014.pdf