C964: Computer Science Capstone
Automatic Content Moderation System

Xavier Alejandro Loera Flores

ID:011037676

xloeraf@wgu.edu

Western Governors University

C964 Computer Science Capstone

05/23/2024

Letter of Transmittal

03/20/2024
Miguel Long Taplin
The Social Club L.L.C.
1600 Amphitheatre Parkway, Mountain View, CA 94043

Dear Miguel Long Taplin,

I am writing to you today to propose a potential solution to a problem plaguing The Social Club L.L.C.'s platform. The Social Club platform is currently experiencing a high volume of user traffic and in particular, there has been a significant increase in the amount of user-generated content, especially content that goes against the platform's moderation policies. This has led to an increase in the amount of moderation request tickets to be processed by the moderation team resulting in a backlog of tickets and a decrease in the quality of moderation on the platform.

To address this issue, I propose implementing a machine learning-based content moderation system that can automatically detect and flag inappropriate content. The machine learning model will be trained on large datasets of data that have been labeled for racist or explicit content. Using supervised learning and natural language processing techniques, the model will be able to learn patterns in the data and make predictions on new data to flag content that is likely to be inappropriate. The main objective of the model would be to moderate new posts as users submit them, providing immediate feedback to the user about the visibility of their posts. This would help reduce the amount of inappropriate content that is visible on the platform and improve the overall user experience. The model will be implemented using logistic regression and will be trained on labeled and flagged post-data. This will all be done in accordance with the platform's moderation policies as well as any data privacy policies and regulations that may apply. Users would also have the ability to opt out of any data collection for training purposes.

The machine learning-based content moderation system would be beneficial to The Social Club platform in several ways. Firstly, it would help reduce the workload of the moderation team by automatically flagging inappropriate content, allowing the moderation team to focus on reviewing flagged content that has been appealed by the poster. Secondly, it would help improve the quality of moderation on the platform by providing consistent and accurate moderation decisions. Finally, it would help improve the user experience on the platform by providing immediate information about the visibility of their posts.

My team and I all have several years of experience in deploying end-to-end machine-learning solutions within the company. Our team plans on adhering to a waterfall project management methodology to develop and implement the system with an estimated timeline of 3 months for development. Our team consists primarily of full-stack engineers as well as dedicated machine learning engineers resulting in a cost of about $350k-$180k for development assuming we meet a 3 month timeline. Once we are complete with the project, the company will then maintain the system with an IT team resulting in an ongoing cost of about $10k-$15k per month for maintenance. We are well equipped to handle the task at hand and are confident that we can deliver a high-quality solution that meets the needs of The Social Club L.L.C. We are excited about the opportunity to implement the solution as outlined and look forward to discussing this proposal further.

Sincerely,

Xavier Loera Flores

Xavier Loera Flores, Software Engineer

Xavier Loera Flores

Project Recommendation

Problem Summary

The Social Club L.L.C's social media platform is currently facing numerous issues that are hindering growth, hurting user experiences, and costing the company vast amounts of resources. Large amounts of user-generated content are being uploaded daily at an exponential rate which consequently means more content that goes against the platform's moderation policies is also being uploaded as well. The company's moderation efforts are unable to keep pace with the increase in the amount of moderation request tickets to be processed by the moderation team. As a result, the platform is facing a backlog of moderation ticket requests, and a decrease in the quality of moderation for posts on the platform. While the company can attempt to scale the moderation team to meet the demand, this would be costly and inefficient. The company needs a more scalable and efficient solution to address the issue of inappropriate content on the platform.

Application Benefits

The machine learning content moderation system will benefit the Social Club L.L.C. in several ways. Firstly, it will help reduce the workload of the moderation team by automatically identifying inappropriate content, allowing the moderation team to focus on reviewing appealed content rather than addressing every report ticket. Secondly, it will help improve the quality of moderation on the platform by providing consistent and accurate moderation decisions in an instant without needing a user to report the content. Finally, it will help improve the user experience on the website by allowing users to have vital content filtering information before they post. Once implemented, there should be a reduction in the amount of publicly visible inappropriate content on the platform and a lower amount of strain on the moderation team.

Application Description

The application will be a full-stack web application that allows users to make posts to the timeline feed. Every post will be automatically moderated for offensive content the moment the user makes the post. All users will be able to see the public feed and see if posts were marked for offensive content. Users will also be allowed to explore all previous posts to see how these posts were flagged by the machine learning model. Additionally, users will be able to preview whether their posts are offensive without making their content public. At first, most users will not be able to read the contents of offensive posts but can opt-in to gain the ability to view offensive content at their discretion.

Data Description

The machine learning model will be trained on data that will be gathered from data stored on the company's platform. The data will consist of content posted by users that has been categorized for offensive content. Each data point will consist only of content in the social media post, the identifier of the post, and a label indicating whether the content is safe or offensive. For the purposes of the project demo, data will be sourced from 2 datasets from Kaggle containing over 50 thousand posts combined that have been categorized based on content. All of the posts are text only and none of the posts contain any personal information or metadata that could be used to identify individuals who posted the content.

Objectives & Hypothesis

The main objective of the model will be to categorize new posts for moderation, allowing a large portion of moderation to be automated. Once implemented there should be a reduction in the amount of publicly visible inappropriate content on the platform resulting in an improved user experience. The model will be implemented using logistic regression and unsupervised training on labeled data. Hypothetically, the model will be able to accurately categorize new posts based on the patterns in the data that it has learned during training. The model will be evaluated based on its accuracy in categorizing new posts and its ability to increase the efficiency of the moderation team.

Methodology

Project Management Methodology

Since the goal is to implement a solution that includes a model trained with supervised data, the development of the model will adhere to a waterfall style of project management. This is because the project has a clear goal and requirements that can be defined at the beginning of the project and do not require frequent changes. Once the project has been deployed and released, we can revisit potentially switching to an agile approach if we plan to continue improving the model with new features and updates. The waterfall approach will allow the development team to focus on the implementation of the model and ensure that it meets the requirements of the project.

Standard Methodology

Cross-Industry Standard Process for Data Mining, also known as (CRISP-DM), will be used as the standard methodology for the project since it is a widely used and accepted methodology for machine learning projects and provides a structured approach for proceeding with the project.

CRISP-DM(1) is a 6-phase process that includes the following phases:

Business Understanding: The Social Club L.L.C. needs a machine learning solution to help reduce the workload of the moderation team and to improve the quality of the moderation for users on the platform. This would not only save time and resources for the company but would also drive growth and user engagement on the platform.

Data Understanding: The data used to train the model will need to be user-generated content to allow the model to learn patterns in the data and make predictions on new data for offensive content detection.

Data Preparation: The data will need to be cleaned, categorized, and preprocessed before it can be used for modeling. Data will be categorized based on the content within the posts to be used for training the model. The data will also be cleaned to remove common stop words, symbols, and other irrelevant data that may not be useful for the model.

Modeling: Using logistic regression and supervised learning, the model will be trained to accurately categorize the posts based on the offensiveness of the content. This will allow the model to predict new posts and categorize whether they are offensive or not.

Evaluation: The solution will be evaluated to ensure that it meets the requirements and objectives of the project. This will involve testing the model with new data to ensure that it is accurately categorizing the content and saving the human moderation team time.

Deployment: The machine learning solution will be deployed as a backend service that will be utilized within a full-stack application where users can post content under the moderation of the model.

Budget

The following section represents the proposed estimated budget for the project including a cost breakdown for development and maintenance. Labor costs are separated into the categories of engineering labor, project management labor, and maintenance labor. A consolidated hardware and resources cost is also included as well as the ongoing estimated cost for cloud usage. A full breakdown of the labor cost is provided in this budget proposal.

Development Cost Overview Breakdown:

Resource	**Cost**
Engineering Labor	$100K-150K
Project Management Labor	$20K-30K
Hardware & Resources Cost	$30k
Upfront Cost Total	**~$150K-200K**

Ongoing Cost Breakdown:

Resource	**Cost**
Cloud Server Usage	~$35K-45K Monthly
Maintenance Engineering Labor	$10K-15K Monthly
On-Going Total	**~$45K-60k Monthly**

The following labor cost full breakdown estimates are based on the salaries of the individual respective roles. The real cost is calculated based on a 3-month development timeline.

Labor Cost Breakdown:

Type	Role	Salary	Real Cost
Engineering Labor	Machine Learning Engineer	$100K - $150K	$25k - $37.5K
Engineering Labor	Machine Learning Engineer	$100K - $150K	$25k - $37.5K
Engineering Labor	Fullstack Engineer	$100K - $150K	$25k - $37.5K
Engineering Labor	Backend Engineer	$100K - $150K	$25k - $37.5K

Management Labor	Project Manager	$80K - $120K	$20k - $30K
Maintenance Labor	Senior IT Analyst	$84K - $126K	$7.0K - $10.5K Monthly
Maintenance Labor	IT Analyst	$50K - $75K	$1.5k - $2.25K Monthly
Maintenance Labor	IT Analyst	$50K - $75K	$1.5k - $2.25K Monthly

Timeline

A rough estimate of the project timeline is as follows:

Sprint Timeline Overview:

Sprint	Dates	Task
1	04/01/2024 - 04/08/2024	Project Planning
2	04/08/2024 - 04/22/2024	Data Collection
3	04/22/2024 - 05/06/2024	Data Preparation
4	05/06/2024 - 05/27/2024	Model Training
5	05/27/2024 - 06/10/2024	Integration & Testing
6	06/10/2024 - 06/17/2024	Evaluation
7	06/17/2024 - 07/01/2024	Model Deployment

Sprint 1 Project Planning: Define the project requirements such as scope, goals, budget, timeline, and team

Sprint 2 Data Collection: Identify collect, and store the data needed for the project

Sprint 3 Data Preparation: Clean, format, and validate the data to prepare for model training

Sprint 4 Model Training: Set up, train, debug, optimize, and validate the machine learning model

Sprint 5 Integration & Testing: Integrate the model with the platform and develop a testing/maintenance plan

Sprint 6 Evaluation: Monitor and collect feedback from a small subset of users with access to the model for evaluation

Sprint 7 Model Deployment: Release the feature to all users, and begin monitoring, surveying, and documenting the system

Data Precautions

The data used for training the model will be carefully selected to adhere to company policies and legal regulations. Beyond ensuring that data will be collected ethically in accordance with the company's privacy policy, the data will also be anonymized to protect the privacy of the users. During the cleaning process, content in posts will be altered to remove any mentions of other users as well as any personal information that could be used to identify individuals. It is also important to note that we only used publicly available data and did not use any private data in the training of the model. The machine learning model will not use any metadata or personal information to make predictions on the content of the posts. Predictions are made purely based on the content of the posts and the patterns in the data that the model has learned during training.

Expertise

The team is composed of experienced and versatile engineers with strong backgrounds in their respective disciplines. The team comprises 2 machine learning engineers, 1 full-stack engineer, 1 dedicated backend engineer, and a project manager all with high-level degrees in computer science or data science and at least 3 years of experience delivering end-to-end machine learning solutions at large scale. The team has worked on similar projects in the past and has the expertise to deliver a high-quality machine-learning solution that meets the requirements of the project. Every member of the team including myself is dedicated to the success of the project and will work diligently to ensure that the project is delivered on time and within budget.

Project Proposal Plan

Project Summary

Problem Description

The Social Club L.L.C. is experiencing significant challenges with moderating user-generated content on its social media platform. The exponential increase in content has led to an overwhelming number of moderation requests, resulting in a backlog and a decline in moderation quality. Scaling the moderation team is not a feasible solution due to high costs and inefficiency. Thus, the company requires a more scalable and efficient method to manage inappropriate content and improve user experience.

Client Summary and Needs

The Social Club L.L.C. is a social media company that needs an automated content moderation solution to handle the vast amounts of user-generated content. The primary goals include:

- Reducing the workload on the moderation team.
- Improving the quality and consistency of content moderation.
- Enhancing user experience by minimizing exposure to inappropriate content.
- Providing users with content filtering options before posting.

Deliverables

Data Handling and Processing

- Collection and preprocessing of user-generated content data for model training.
- Ensuring data privacy by anonymizing and cleaning the data.

Machine Learning Model

- A logistic regression model trained on labeled data to identify and categorize offensive content.
- The model will be integrated into the backend of the web application to provide real-time moderation.

Full-Stack Web Application

- A platform for users to post content to a timeline feed.
- Automatic moderation of posts for offensive content upon submission.
- Public feed displaying content with moderation indicators.
- Functionality for users to preview the potential offensiveness of their posts.
- Opt-in feature for users to view flagged content.

Documentation

- A comprehensive user guide detailing how to use the application and its features.
- Technical documentation for the development and maintenance of the machine learning model and application.

Benefits Justification

The machine learning content moderation system will significantly benefit The Social Club L.L.C. by:

Efficiency: Automating the moderation process reduces the workload on the moderation team, allowing them to focus on reviewing appeals.

User Experience: Ensuring consistent, fast, and accurate moderation decisions improves the overall quality of content on the platform by hiding content as it's uploaded quickly, filtering out inappropriate content, and providing users with pre-posting feedback on potential content issues.

Scalability: Offering a scalable solution that can handle the growing amount of content without proportional increases in moderation costs or the need to mass hire new team members.

Cost-Effectiveness: By reducing the need for extensive human moderation, the company will be saving on operational costs.

By implementing this solution, the Social Club L.L.C. can address its content moderation challenges effectively, improving both platform integrity and user satisfaction.

Data Summary

Data Source and Collection

The machine learning model will be trained on data that will be gathered from data stored on the company's platform. The data will consist of content posted by users that has been categorized for offensive content. For this demo project, we will source our data from two publicly available datasets on Kaggle, which together contain over 50,000 posts combined. These datasets have already been categorized based on content on whether the content is safe or offensive. The data points will consist of the text of the posts, the ID of each post, and a boolean label containing its classification. Importantly, the data will have no personal information or metadata that could identify individuals.

Data Processing and Management

Design Phase: During the design phase, we will define the structure and requirements of our dataset, ensuring it aligns with our model's needs. The following data schema represents how we will store the data:

- text (string): The text content of the social media post
- id (number): A unique identifier
- is_nsfw (boolean): A flag indicating if it is offensive (true) or safe (false)

Development Phase: In the development phase, data preprocessing will be critical. This includes:

- Data Cleaning: Removing mentions of other users and any personal information to ensure anonymity.
- Data Normalization: Standardizing the format of the text (e.g., lowercasing, removing special characters) to maintain consistency.
- Data Splitting: Dividing the dataset into training, validation, and test sets to evaluate the model's performance accurately.

Maintenance Phase: During maintenance, we will continuously monitor the data and model performance. This involves:

- Updating the Dataset: Saving new data posted to the platform to prepare in case we need to retain the model.
- Handling Anomalies: Identifying and managing data anomalies such as outliers or incomplete data.

- Re-training the Model**:** The model can be retrained with the updated dataset to ensure it adapts to evolving patterns in offensive content.

Justification of Data Suitability

The chosen datasets from Kaggle are well-suited for our project due to their large dataset size and categorized data. The data also features no personal information and metadata which ensures that our model's predictions are based solely on the content of the posts and not on other contextual factors. In order to handle data anomalies, we will need to clean and preprocess the data which contains a large amount of words and symbols that are irrelevant. Outliers and incomplete data will be managed through preprocessing steps, ensuring the integrity and quality of the dataset.

Ethical and Legal Considerations

Our data handling and processing protocols are designed to adhere strictly to ethical guidelines, legal regulations, and company policy.

- Privacy: Data will be anonymized by removing personal information and user mentions.
- Ethical Sourcing: Only publicly available data will be used while abiding by our privacy policy.
- Transparency: Clear documentation of data sources and collection practices will be made public.

There are no significant ethical or legal concerns with our data as we will ensure that all data is anonymized and ethically sourced. By focusing solely on publicly available data and removing any potentially identifying information, we comply with privacy laws and ethical standards. The users of the platform can read about the data collection process and opt out of data collection for training purposes. In summary, the data we will source from Kaggle for the demo and the data we will source from our users will ensure that we meet the project's needs effectively while upholding the highest ethical and legal standards.

Implementation

Project Management Methodology

Since the goal is to implement a solution that includes a model trained with supervised data, we will adhere to a waterfall project management approach especially given the project’s clear goals and well-defined requirements. This approach allows us to systematically progress through each development phase, ensuring all specifications are met before moving forward. Once the model is deployed, we may revisit and transition to an agile methodology to accommodate iterative improvements and feature enhancements if the company sees success in the implementation of the service. This initial waterfall approach will ensure the development team can focus on meeting the precise project requirements before iterating on future enhancements.

Standard Methodology

Cross-Industry Standard Process for Data Mining, also known as (CRISP-DM), will be used as the standard methodology for the project since it is widely used for machine learning projects and provides a structured approach for proceeding with the project.(1)

CRISP-DM is a 6-phase process that includes the following phases:

Business Understanding: The Social Club L.L.C. needs an automatic machine learning service to help reduce the workload of the moderation team and to improve the quality of the moderation for users on the platform. This would not only save time and resources for the company but would also drive growth and user engagement on the platform.

Data Understanding: The data used to train the model will need to be user-generated content to allow the model to learn patterns in the data and make predictions on new data for offensive content detection. Using user-generated content in a Natural Language Processing model will allow the model to learn the context and patterns of offensive content.

Data Preparation: The data will need to be cleaned, categorized, and preprocessed before it can be used for training to ensure the model isn't training on incomplete or irrelevant data. Data will be categorized based on the content within the posts to be used for training the model. The data will also be cleaned to remove common stop words, symbols, and other anomalies that may not be useful for the model.

Modeling: Using natural language processing and supervised learning techniques and logistic regression, the model will be trained to accurately categorize the posts based on the offensiveness of the content. This will allow the model to predict future user-generated posts and categorize whether they are offensive or not allowing the content to be automatically moderated.

Evaluation: The solution will be evaluated to ensure that it meets the requirements and objectives of the project. This will involve testing the model with testing data and new user-generated data to ensure that it is accurately categorizing the content and saving the human moderation team time.

Deployment: The machine learning solution will be deployed as a backend service that will be utilized within a full-stack application where users can post content that will be categorized by the model. Users will also be able to see how the model moderates their content.

Timeline

The project timeline forecasts the development process for each milestone step along with the deliverables for each of the milestones. Overall, the project is expected to be completed in about 3 months or about 90 days. The project timeline is as follows:

Sprint Timeline Overview:

Sprint	Dates	Task
1	04/01/2024 - 04/08/2024	Project Planning
2	04/08/2024 - 04/22/2024	Data Collection
3	04/22/2024 - 05/06/2024	Data Preparation
4	05/06/2024 - 05/27/2024	Model Training
5	05/27/2024 - 06/10/2024	Integration & Testing
6	06/10/2024 - 06/17/2024	Evaluation
7	06/17/2024 - 07/01/2024	Model Deployment

The project timeline is broken down into 7 milestones each lasting 1-3 weeks. Each milestone sprint is broken down into the following tasks and deliverables:

Sprint 1: Project Planning (1 Week)

- Define the project scope and goals
- Clearly define the project budget
- Identify and confirm the project team
- Establish the project timeline and assign tasks

Sprint 2: Data Collection (2 Weeks)

- Identify and validate the training data sources
- Plan the data collection method
- Write code scripts to collect the data
- Store the data in a database for preparation

Sprint 3: Data Preparation and Processing (2 Weeks)

- Format the data using the appropriate standardized data structures
- Clean the data to remove any irrelevant information, duplicates, or errors
- Anonymize the data and mitigate data anomalies
- Validate the data and prepare it for training

Sprint 4: Model Training (3 Weeks)

- Set up the machine learning environment
- Identify and clarify vectorization technique and machine learning algorithm
- Split and vectorize the data
- Develop and train the machine learning model
- Debug and optimize the model
- Validate the vectorizer and model and save them for integration

Sprint 5: Integration & Testing (2 Weeks)

- Integrate the vectorizer and model with the platform
- Write tests for the model and the backend service
- Test the model and the service
- Implement end-to-end testing
- Create a maintenance plan for the IT team

Sprint 6: Evaluation (1 Week)

- Prepare the model for deployment and create feature flags
- A/B test the model with a small subset of users
- Monitor the system and collect feedback
- Evaluate the solution and document the results

Sprint 7: Model Deployment (2 Weeks)

- Deploy the model to the entire platform
- Monitor the system and collect feedback
- Document the system and write post-implementation reports

Evaluation Plan

At each stage of the project, we will evaluate the progress of the machine learning model and the full-stack application to verify that the project is on track and meeting the requirements of the Social Club L.L.C. Alongside requirement validation and extensive testing throughout development the following metrics are going to be tracked:

- Model Accuracy: The model's accuracy will be evaluated using a testing dataset to ensure that it is accurately categorizing the content.
- Model Efficiency: The model's efficiency will be evaluated by measuring the time it takes to process a large number of requests.

Since the project's main goals are to improve customer satisfaction and reduce the workload of the moderation team, the verification methods will focus on the model's accuracy and efficiency. The best model to solve the problem of the Social Club L.L.C.'s problem will be the one that can process requests as fast as possible while being as accurate as possible. The models will be benchmarked to measure the model's accuracy on a testing dataset and the model's speed on a large number of requests. Post implementation, verification methods will include A/B testing on users, user feedback, and monitoring the system for any issues.

Resources and Costs

The following section represents the proposed estimated budget for the project including a cost breakdown for development and maintenance. Labor costs are separated into the categories of engineering labor, project management labor, and maintenance labor. A consolidated hardware and resources cost is also included as well as the ongoing estimated cost for cloud usage. A full breakdown of the labor cost is provided in this budget proposal.

Development Cost Overview Breakdown:

Resource	**Cost**
Engineering Labor	$100K-150K

Project Management Labor	$20K-30K
Database Storage	$5,000
Data Processing Compute Usage	$20,000
Upfront Cost Total	**~$150K-200K**

Ongoing Cost Breakdown:

Resource	**Cost**
Cloud Server Usage	~$35K-45K Monthly
Maintenance Engineering Labor	$10K-15K Monthly
On-Going Total	**~$45K-60k Monthly**

The following labor cost full breakdown estimates are based on the salaries of the individual respective roles. The real cost is calculated based on a 3-month development timeline.

Labor Cost Breakdown:

Type	Role	Salary	Real Cost
Engineering Labor	Machine Learning Engineer	$100K - $150K	$25k - $37.5K
Engineering Labor	Machine Learning Engineer	$100K - $150K	$25k - $37.5K
Engineering Labor	Fullstack Engineer	$100K - $150K	$25k - $37.5K
Engineering Labor	Backend Engineer	$100K - $150K	$25k - $37.5K
Management Labor	Project Manager	$80K - $120K	$20k - $30K
Maintenance Labor	Senior IT Analyst	$84K - $126K	$7.0K - $10.5K Monthly
Maintenance Labor	IT Analyst	$50K - $75K	$1.5k - $2.25K Monthly
Maintenance Labor	IT Analyst	$50K - $75K	$1.5k - $2.25K Monthly

Application

Overview

The web application allows users to view and create posts to a public timeline that is automatically moderated using machine learning to classify posts for offensive content. Users will have the ability to view how the model classifies their posts and the posts of others. The application also features pages dedicated to giving an in-depth analysis of models and data where users can gain a better understanding of the machine learning moderation system.

Links

- Frontend : https://wgu-capstone-xavier-loera-flores.vercel.app
- Backend : https://wgu-capstone-production.up.railway.app/
- Live Docs : https://wgu-capstone-docs.vercel.app/

Files

- model: SciKit-Learn Machine Learning Model Python Files
- backend: FastAPI Python Server
- frontend: NextJS Web Application
- database: Drizzlekit PostgreSQL Database
- data: Seed Data for PostgreSQL Database
- docs: Insomnia Generated API Documentation
- project_management: Project Management Documentation
- insomnia.json: Insomnia Project File
- README.md: Project README File
- .gitignore: Git Ignore File

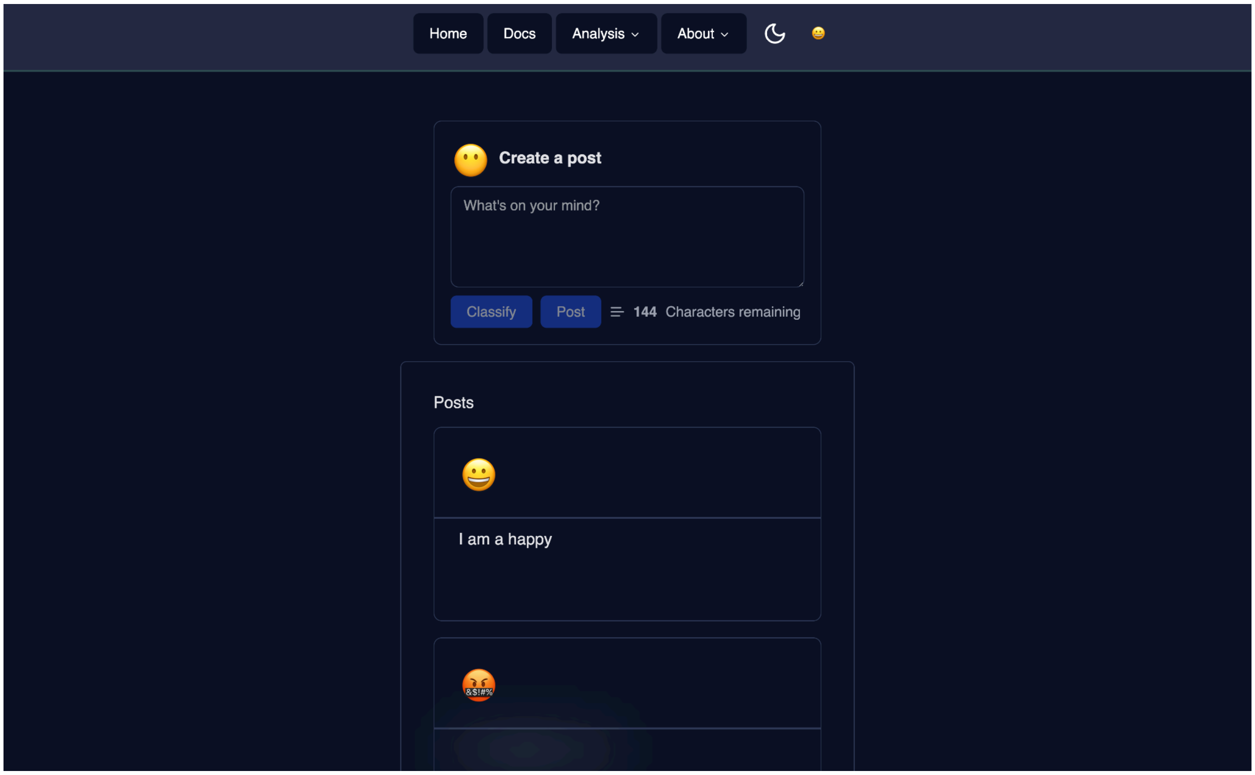
Machine Learning Algorithm & Vectorizer

The machine learning algorithm used in this project is the Logistic Regression and the vectorizer method used in this project is the TF-IDF(Term Frequency Inverse Document Frequency) both of which have implementations in the SciKit-Learn library.

User Interface

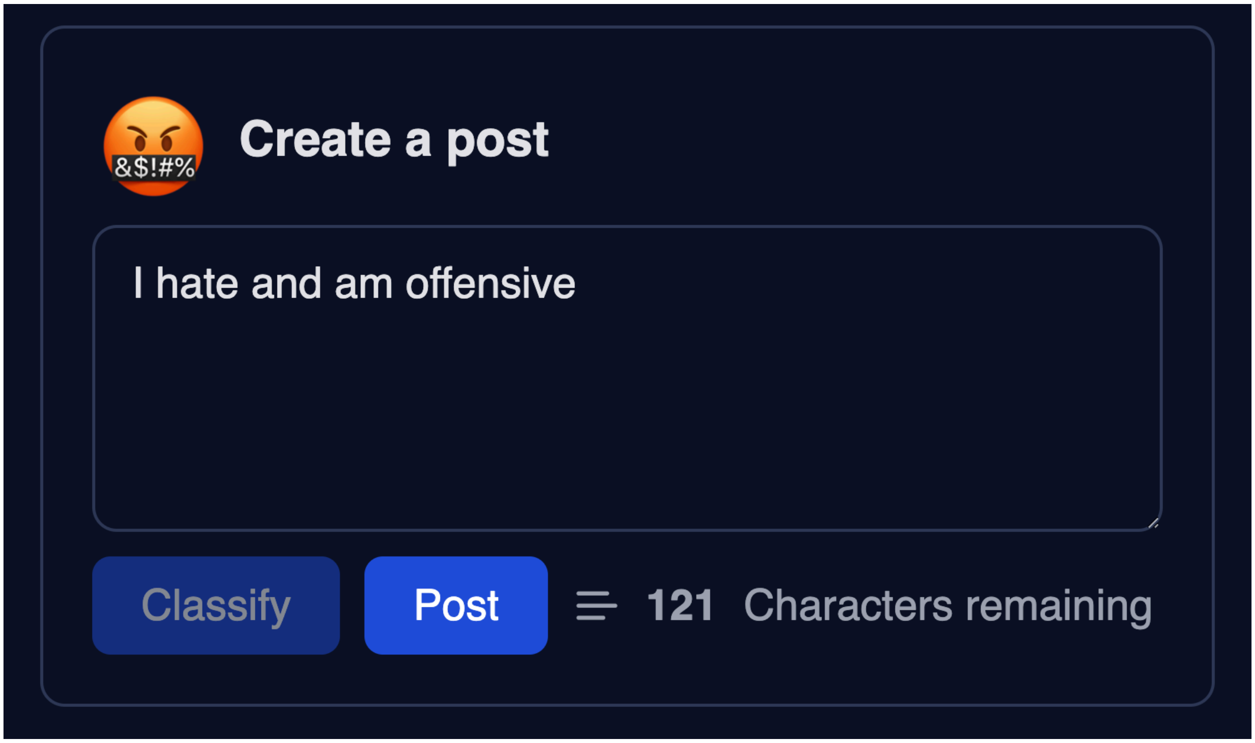
The user interface is built using the NextJS framework which is a ReactJS framework. The user interface is designed to be simple and responsive to work on all devices. The following are screenshots of key components of the user interface.

1. Home Page with Post Feed



I am a happy

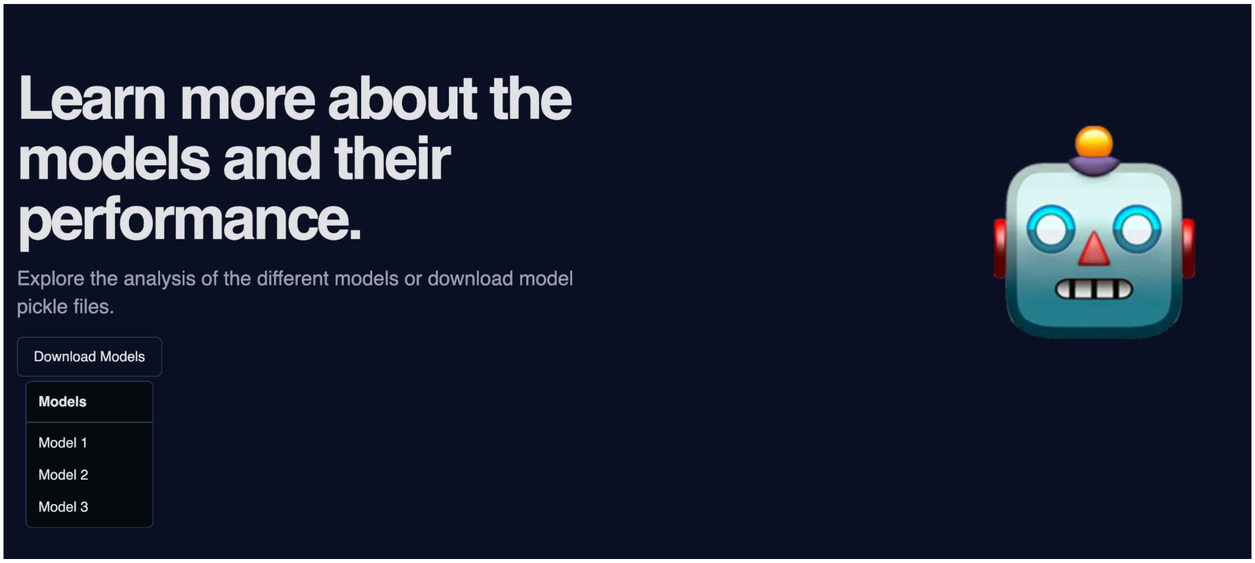
2. Post Composer with Classification



Create a post

I hate and am offensive

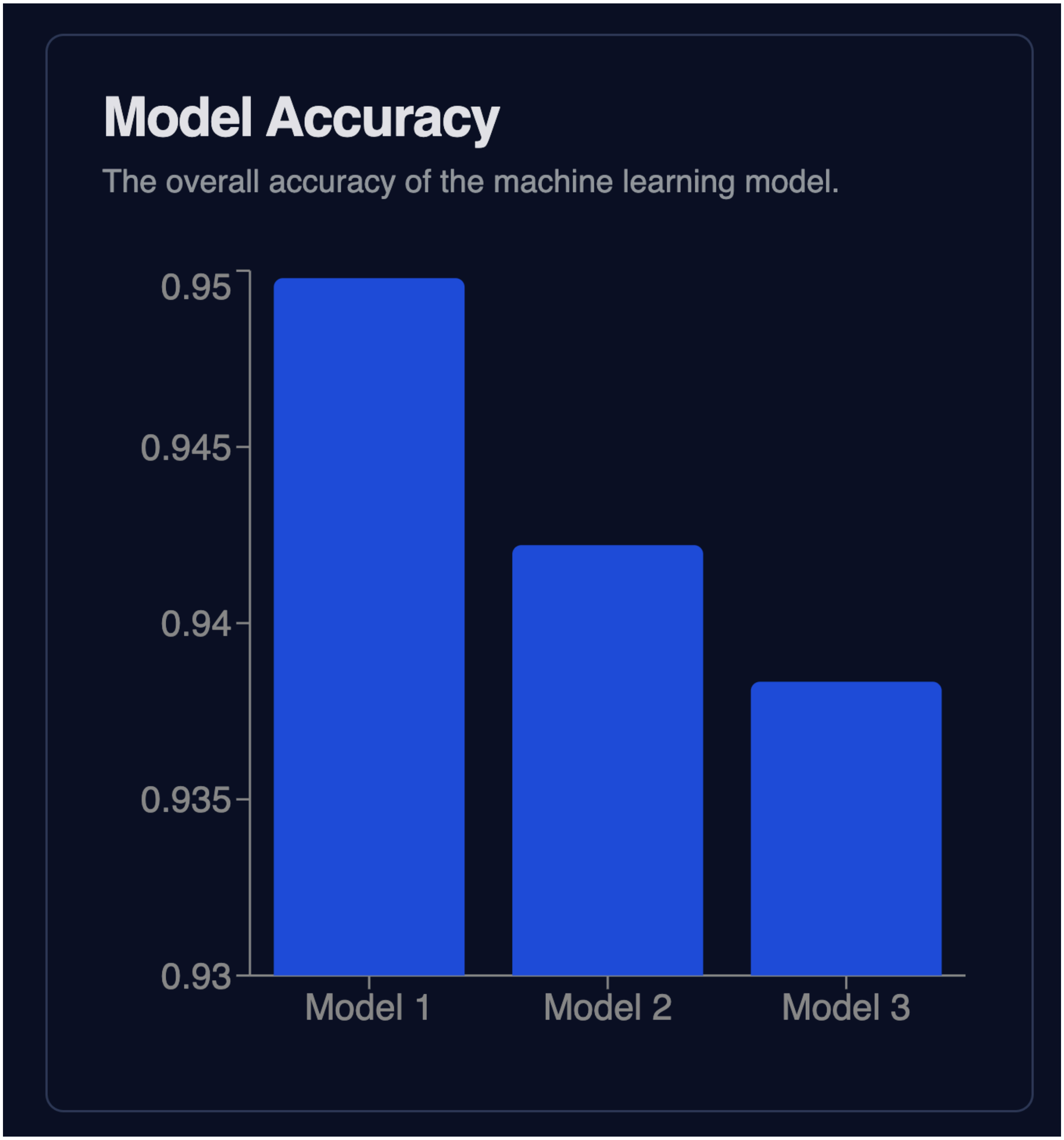
3. Analysis Header



Learn more about the models and their performance.

Explore the analysis of the different models or download model pickle files.

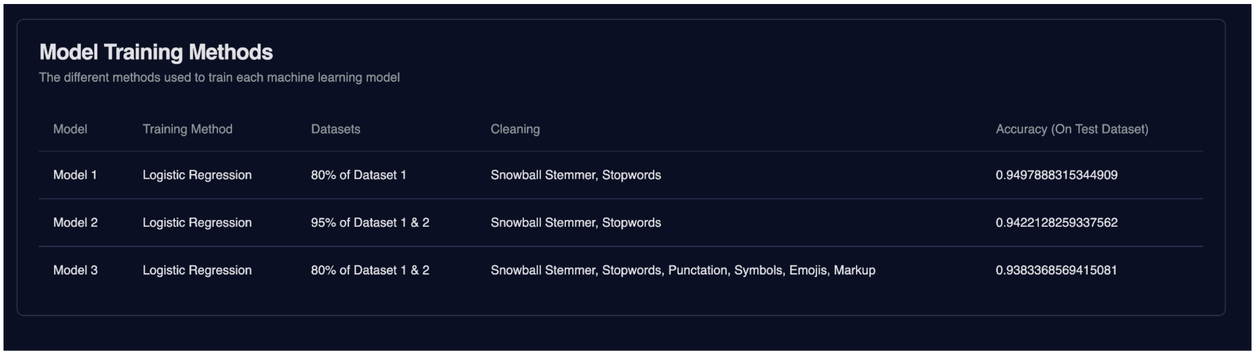
4. Analysis Card with Data Chart



Model Accuracy

The overall accuracy of the machine learning model.

5. Data Table

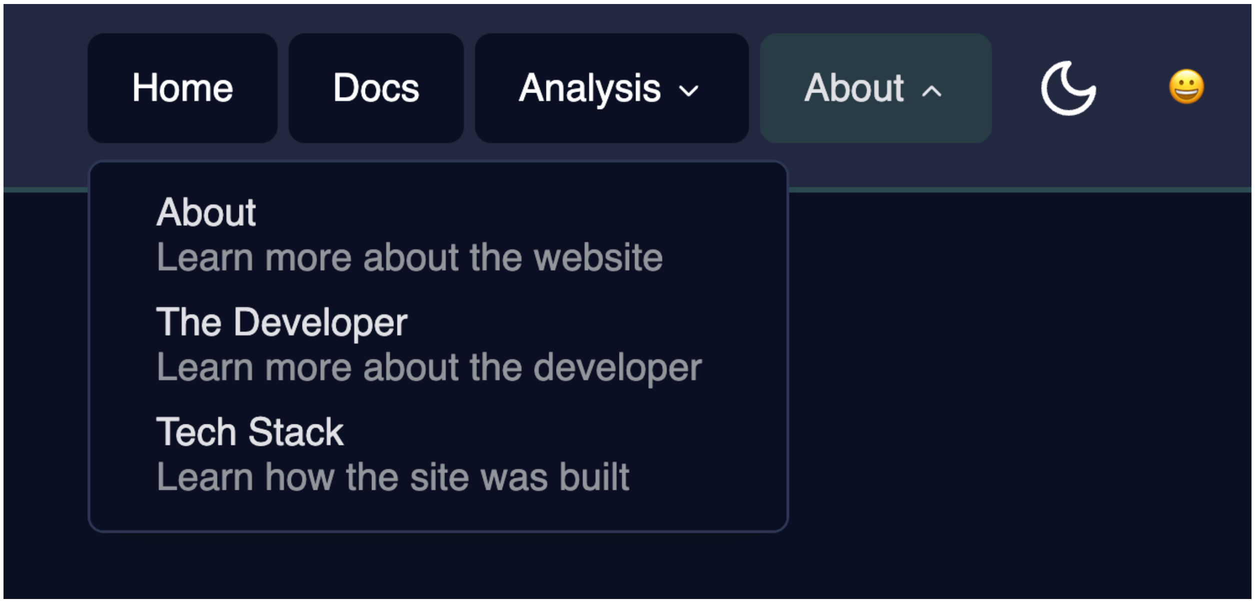


Model Training Methods

The different methods used to train each machine learning model

Model	Training Method	Datasets	Cleaning	Accuracy (On Test Dataset)
Model 1	Logistic Regression	80% of Dataset 1	Snowball Stemmer, Stopwords	0.9497888315344909
Model 2	Logistic Regression	95% of Dataset 1 & 2	Snowball Stemmer, Stopwords	0.9422128259337562
Model 3	Logistic Regression	80% of Dataset 1 & 2	Snowball Stemmer, Stopwords, Punctation, Symbols, Emojis, Markup	0.9383368569415081

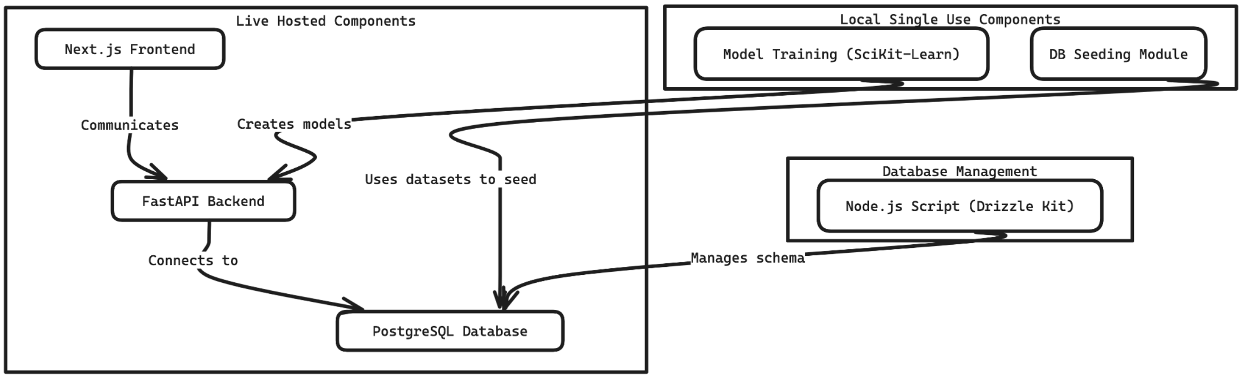
6. Navigation Header


Home
Docs
Analysis
About
About
Learn more about the website
The Developer
Learn more about the developer
Tech Stack
Learn how the site was built

System Design

The live hosted components of the application consist of a Next.js frontend hosted on Vercel which communicates with a FastAPI backend hosted on Railway which itself connects to a PostgreSQL database hosted on Vercel. The other components of the system are hosted locally on the developer's machine and are run manually when needed. The PostgreSQL database is managed by a Node.js script using Drizzle Kit to manage the database schema. The backend server uses models created and saved from a Python script that uses the SciKit-Learn library to train a machine learning model. There is also a seeding module that uses the saved models to classify and seed the database with posts.

System Design Diagram:



Post-implementation Report

Solution Summary

The Social Club L.L.C. faced significant challenges in moderating the vast amount of user-generated content on its social media platform, resulting in a backlog and a decline in moderation quality. To address these issues, the company needed an efficient and scalable content moderation system. The implemented solution is a machine learning-based automated moderation system that reduces the workload on the moderation team, provides consistent and fast content filtering, and enhances user experience by reducing inappropriate content on the platform.

The new system uses a logistic regression model trained on user-generated data to categorize offensive content. Since the model is integrated into the backend of the social media platform, it enables real-time content moderation and significantly improves the efficiency of the moderation process. Additionally, the solution includes a full-stack web application that allows users to post and view auto-moderated content, preview potential content warnings before posting, and opt-in to view inappropriate content. This approach ensures the scalability of the moderation process and also offers a cost-effective way to maintain high-quality content on the platform.

The Social Club L.L.C. can handle the increasing volume of content without expanding the moderation team and save on long-term operational costs by automating content moderation. The machine learning system is supported by a responsive and mobile-friendly interface built with the Next.js framework that can be accessed on most modern web browsers. The deployment architecture is robust and scalable since it features a Next.js frontend hosted on Vercel, a FastAPI backend hosted on Railway, and a PostgreSQL database hosted via VercelDB all of which can automatically scale with traffic. In summary, the implemented solution ensures a high-quality user experience and reduces the workload on the moderation team by providing a cost-effective, accurate, and efficient way to moderate content on the Social Club L.L.C. platform.

Data Summary

The data was sourced from two separate datasets from Kaggle. The first dataset, the training data from the Twitter Sentiment Analysis(2) dataset(train.csv) contained over 29 thousand rows of data containing a social media post and a label indicating whether the post was racist or not. The second dataset, the Hate Speech & Offensive Language(3) dataset(labeled_data.csv), contained over 25 thousand rows of data with a social media post and a label indicating whether the post contained hate speech, offensive language, or neither. The data was processed to map the labels to a binary classification of offensive or not offensive. The data was then split into training and testing sets for use in the machine learning model.

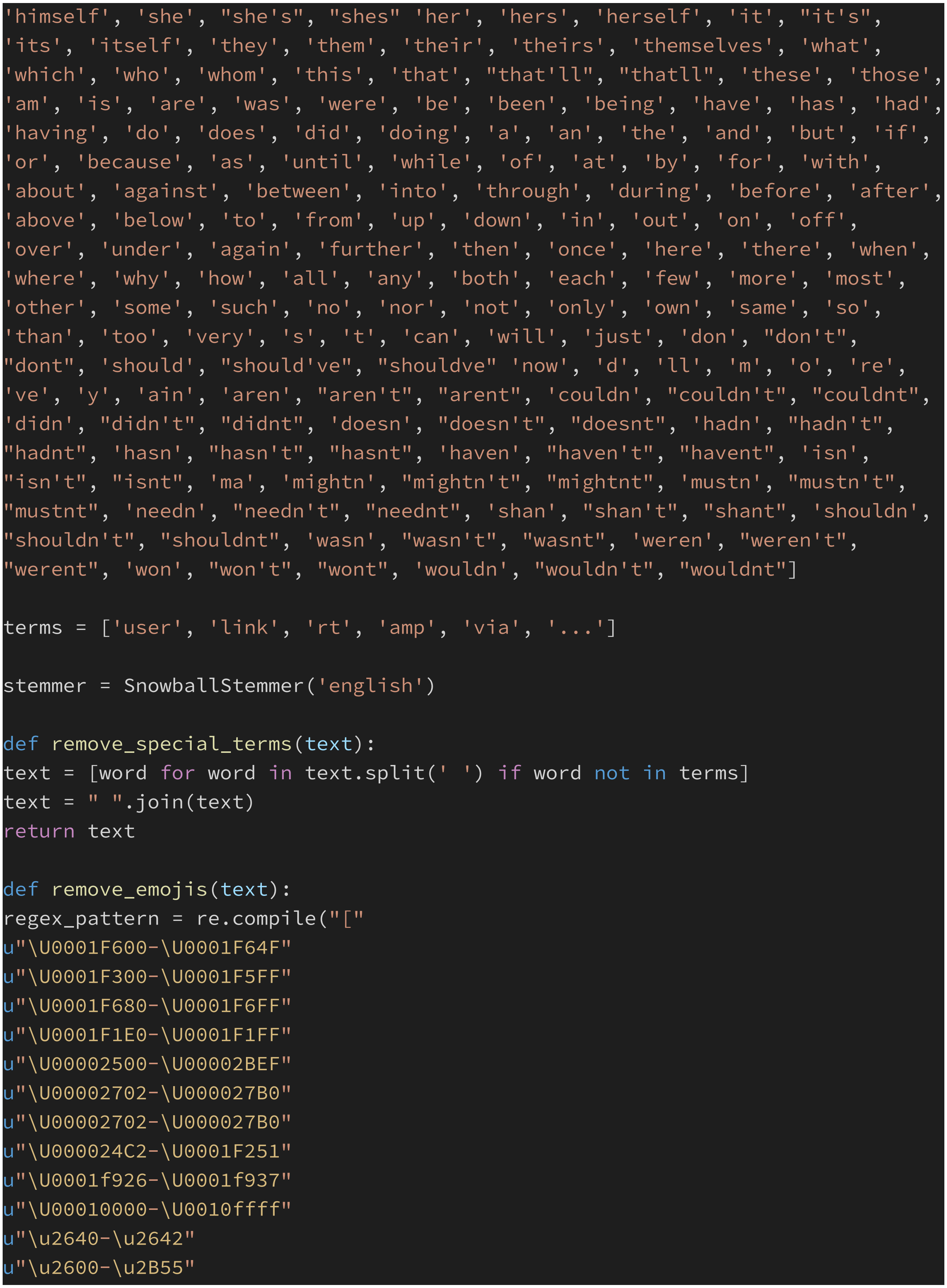
data.py

```
from pandas import read_csv, concat
from clean import clean_data

def preprocess_data():
data = read_csv('../datasets/train.csv')
data2 = read_csv('../datasets/labeled_data.csv')
X = concat([data['tweet'], data2['tweet']])
X = X.apply(clean_data)
raw_y1 = data['label']
raw_y2 = data2['class']
categorized_y1 = raw_y1.map({1: 'offensive', 0: 'safe'})
categorized_y2 = raw_y2.map({0: 'offensive', 1: 'offensive', 2: 'safe'})
y = concat([categorized_y1, categorized_y2])
return X, y

def preprocess_test_data():
data = read_csv('../datasets/test.csv')
X = data['tweet']
X = X.apply(clean_data)
return X
```

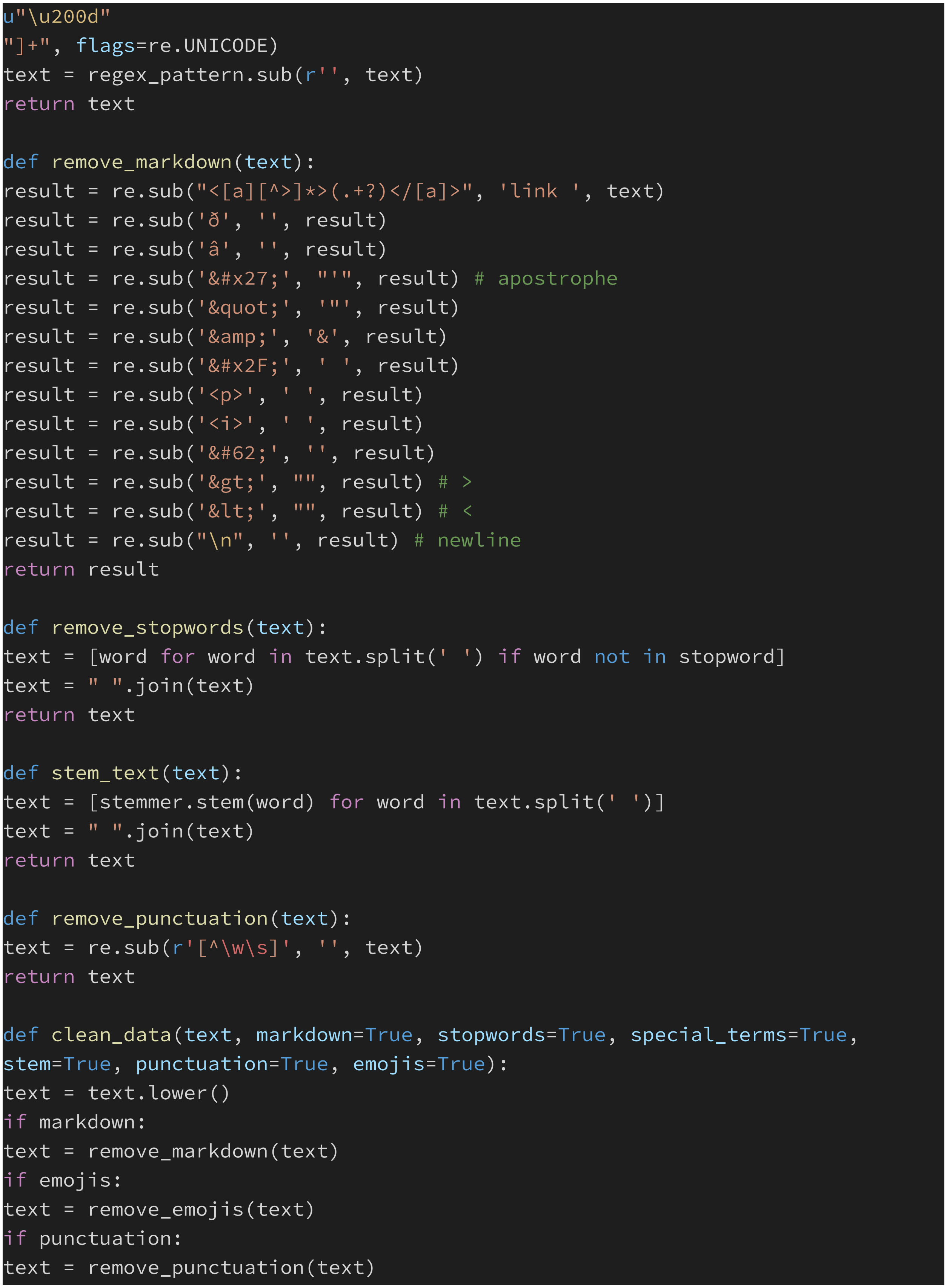
The data was sourced from CSV files and loaded into pandas for processing. After loading the different datasets, the data was then preprocessed to standardize the format of the data from the different datasets. The data was then combined into a singular dataset that was cleaned and prepared for splitting into training and testing sets.

clean.py

```
from nltk import SnowballStemmer
import re

stopword = ['i', 'u', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves',
'you', "you're", "youre", "you've", "youve", "you'll", "youll", "you'd",
"youd",'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his',
```



```
'himself', 'she', "she's", "shes" 'her', 'hers', 'herself', 'it', "it's",
'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what',
'which', 'who', 'whom', 'this', 'that', "that'll", "thatll", 'these', 'those',
'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had',
'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if',
'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with',
'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after',
'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off',
'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when',
'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most',
'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so',
'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't",
"dont", 'should', "should've", "shouldve" 'now', 'd', 'll', 'm', 'o', 're',
've', 'y', 'ain', 'aren', "aren't", "arent", 'couldn', "couldn't", "couldnt",
'didn', "didn't", "didnt", 'doesn', "doesn't", "doesnt", 'hadn', "hadn't",
"hadnt", 'hasn', "hasn't", "hasnt", 'haven', "haven't", "havent", 'isn',
"isn't", "isnt", 'ma', 'mightn', "mightn't", "mightnt", 'mustn', "mustn't",
"mustnt", 'needn', "needn't", "neednt", 'shan', "shan't", "shant", 'shouldn',
"shouldn't", "shouldnt", 'wasn', "wasn't", "wasnt", 'weren', "weren't",
"werent", 'won', "won't", "wont", 'wouldn', "wouldn't", "wouldnt"]

terms = ['user', 'link', 'rt', 'amp', 'via', '...']

stemmer = SnowballStemmer('english')

def remove_special_terms(text):
text = [word for word in text.split(' ') if word not in terms]
text = " ".join(text)
return text

def remove_emojis(text):
regex_pattern = re.compile("["
u"\U0001F600-\U0001F64F"
u"\U0001F300-\U0001F5FF"
u"\U0001F680-\U0001F6FF"
u"\U0001F1E0-\U0001F1FF"
u"\U00002500-\U00002BEF"
u"\U00002702-\U000027B0"
u"\U00002702-\U000027B0"
u"\U000024C2-\U0001F251"
u"\U0001f926-\U0001f937"
u"\U00010000-\U0010ffff"
u"\u2640-\u2642"
u"\u2600-\u2B55"
```



```
u"\u200d"
"]+", flags=re.UNICODE)
text = regex_pattern.sub(r'', text)
return text

def remove_markdown(text):
result = re.sub("<[a][^>]*>(.+?)</[a]>", 'link ', text)
result = re.sub('ð', '', result)
result = re.sub('â', '', result)
result = re.sub('&#x27;', "'", result) # apostrophe
result = re.sub('&quot;', '"', result)
result = re.sub('&amp;', '&', result)
result = re.sub('&#x2F;', ' ', result)
result = re.sub('<p>', ' ', result)
result = re.sub('<i>', ' ', result)
result = re.sub('&#62;', '', result)
result = re.sub('&gt;', "", result) # >
result = re.sub('&lt;', "", result) # <
result = re.sub("\n", '', result) # newline
return result

def remove_stopwords(text):
text = [word for word in text.split(' ') if word not in stopword]
text = " ".join(text)
return text

def stem_text(text):
text = [stemmer.stem(word) for word in text.split(' ')]
text = " ".join(text)
return text

def remove_punctuation(text):
text = re.sub(r'[^\w\s]', '', text)
return text

def clean_data(text, markdown=True, stopwords=True, special_terms=True,
stem=True, punctuation=True, emojis=True):
text = text.lower()
if markdown:
text = remove_markdown(text)
if emojis:
text = remove_emojis(text)
if punctuation:
text = remove_punctuation(text)
```

```
if special_terms:
text = remove_special_terms(text)
if stopwords:
text = remove_stopwords(text)
if stem:
text = stem_text(text)
return text
```

To develop the model, the data was first vectorized using a TF-IDF vectorizer. The vectorized data was then used to train a logistic regression model. After evaluating the model using the testing data to determine its accuracy, the model was then saved to a PKL file for use in the backend server application.

persistence.py

```
import pickle
version = 'v3'


def dump_model(model):
pickle.dump(model, open(f'model-{version}.pkl', 'wb'))


def dump_vectorizer(vectorizer):
pickle.dump(vectorizer, open(f'vectorizer-{version}.pkl', 'wb'))


def load_model():
return pickle.load(open(f'model-{version}.pkl', 'rb'))


def load_vectorizer():
return pickle.load(open(f'vectorizer-{version}.pkl', 'rb'))


def persist(model, vectorizer):
dump_model(model)
dump_vectorizer(vectorizer)
```

model.py

```
from data import preprocess_data, preprocess_test_data
from visualization import visualize
from test import test_model
from persistence import persist
from train import train

visualize()
X, y = preprocess_data()
test = preprocess_test_data()
model, vectorizer = train(X, y, test)
test_model(model, vectorizer)
persist(model, vectorizer)
```

A separate module then uses those saved models and vectorizers to classify the posts from the dataset. After classifying the posts, the data and classifications are then saved to a PostgreSQL database.

Machine Learning

What: Logistic Regression is a statistical method that can be well suited for binary classification problems. It serves as the project's primary algorithm for identifying and categorizing offensive content in user-generated posts by classifying whether a post is offensive or not.

How: The development of the logistic regression model involved several key steps. First, the user-generated labeled content dataset was preprocessed to clean, stem, and standardize the text by removing stop words, normalizing cases, and using tokenization. The cleaned text data was then vectorized using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer to transform the text into numerical features which were used to train the logistic regression model on the labeled dataset. After evaluating the model on accuracy and efficiency, the trained model was saved and persisted as a PKL (Pickle) file for integration into the backend server application.

Why: Logistic Regression was chosen because of its effectiveness in binary classification and natural language processing problems. The logistic regression implementation from the SciKit-Learn library provides a probabilistic approach that is easy to understand and implement. Additionally, logistic regression is computationally efficient which is suitable for real-time moderation on the server while demonstrating a capability to accurately identify offensive content. Logistic regression aligns with the Social Club L.L.C's goal of providing a reliable, fast, and scalable content moderation system.

The following code snippet shows the implementation of the logistic regression model using the SciKit-Learn library in Python.

train.py

```python
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import time

def print_accuracy_metrics(model, x_test_vec, y_test):
y_pred = model.predict(x_test_vec)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

def print_speed_metrics(model, test_vec):
print("Measuring Time...")
start_time = time.time()
for _ in range(10000):
model.predict(test_vec)
end_time = time.time()
elapsed_time = end_time - start_time
print("Elapsed Time:", elapsed_time)

def fit_vectorizer(x_train, x_test):
vectorizer = TfidfVectorizer()
x_train_vec = vectorizer.fit_transform(x_train)
x_test_vec = vectorizer.transform(x_test)
return vectorizer, x_train_vec, x_test_vec

def fit_model(x_train_vec, y_train):
model = LogisticRegression()
model.fit(x_train_vec, y_train)
return model
```

```
def split_data(x,y):
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.05,
random_state=64)
return x_train, x_test, y_train, y_test

def train(x,y, test):
x_train, x_test, y_train, y_test =split_data(x,y)
vectorizer, x_train_vec, x_test_vec = fit_vectorizer(x_train, x_test)
model = fit_model(x_train_vec, y_train)
print_accuracy_metrics(model, x_test_vec, y_test)
test_vec = vectorizer.transform(test)
print_speed_metrics(model, test_vec)

return model, vectorizer
```

Validation

Upon the completion of the machine learning project for the Social Club L.L.C. project, we carried out a comprehensive validation process as outlined in our project proposal. The primary metrics used for evaluation were model accuracy and processing speed, with the objective of enhancing customer satisfaction and easing the workload of the moderation team. Three models were tested: Model 1, Model 2, and Model 3.

Model Accuracy and Efficiency Evaluation

Model 1 demonstrated the highest accuracy at 0.9498, followed by Model 2 at 0.9422, and Model 3 at 0.9383. While all three models performed well in terms of accuracy, Model 1 outperformed the others, indicating its superior capability in accurately categorizing content. However, upon using the model in real-world tests, it appeared as if Model 3 performed the best from a subjective perspective since the first two models failed to recognize very clear cases of offensive content. In terms of efficiency, Model 1 processed requests at a speed of 0.7006 seconds per request, which is slightly slower than Model 2's 0.6921 seconds but significantly faster than Model 3, which took 4.4584 seconds per request. Although Model 2 was slightly faster than Model 1, the difference was negligible, meaning Model 1 would be the optimal choice given its higher accuracy. Model 3 was however chosen because of its subjective performance.

These validation tests need to be revisited and more data is needed to determine the best model.

Post-Implementation Verification

Following the deployment of the machine learning system, extensive A/B testing will be conducted with a segment of users to compare the user experience of the new system against the previous one. Additionally, continuous monitoring will be needed to ensure there are no significant issues and that the system maintains high performance even under intense loads. The business team will need to consult and measure the efficiency gains in processing speed and accuracy directly contributed to reduced moderation workload.

The following code snippet shows how accuracy and speed were measured and displayed in the validation process.

From train.py

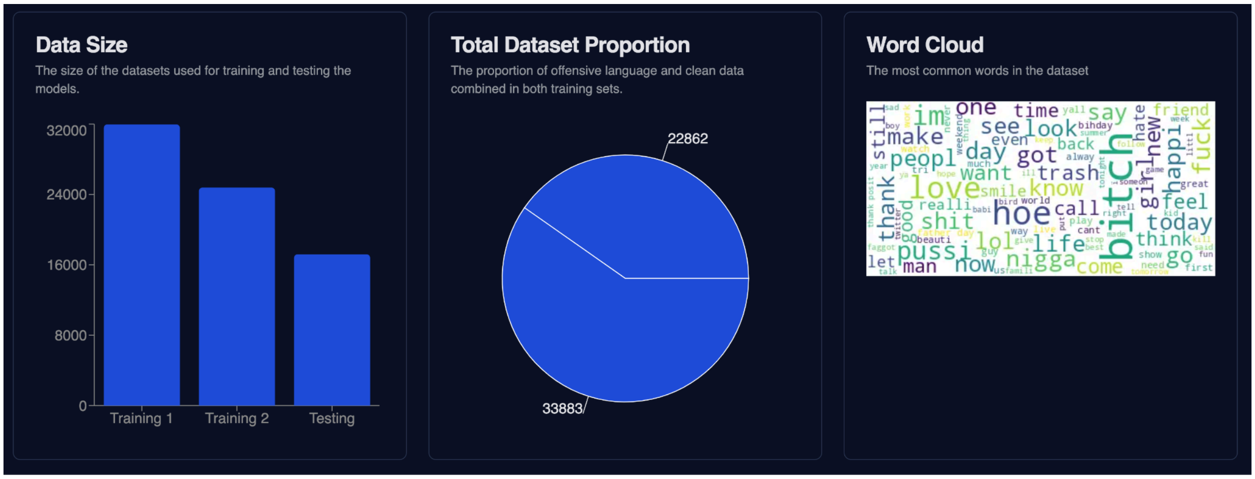
```python
def print_accuracy_metrics(model, x_test_vec, y_test):
y_pred = model.predict(x_test_vec)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

def print_speed_metrics(model, test_vec):
print("Measuring Time...")
start_time = time.time()
for _ in range(10000):
model.predict(test_vec)
end_time = time.time()
elapsed_time = end_time - start_time
print("Elapsed Time:", elapsed_time)
```

Visualizations

Below are visualizations of the data and the machine-learning model. Some of the images are sourced from the development process and some are sourced from the final web application. The web application features high-resolution images and interactive charts on the analysis pages that can be accessed via the navigation bar.

Visualizations 1-3 can be found on the analysis pages of the websites featuring different charts about the models or the data.

Visual 1: Charts about the data with wordcloud(Visual 6)



Data Size

The size of the datasets used for training and testing the models.

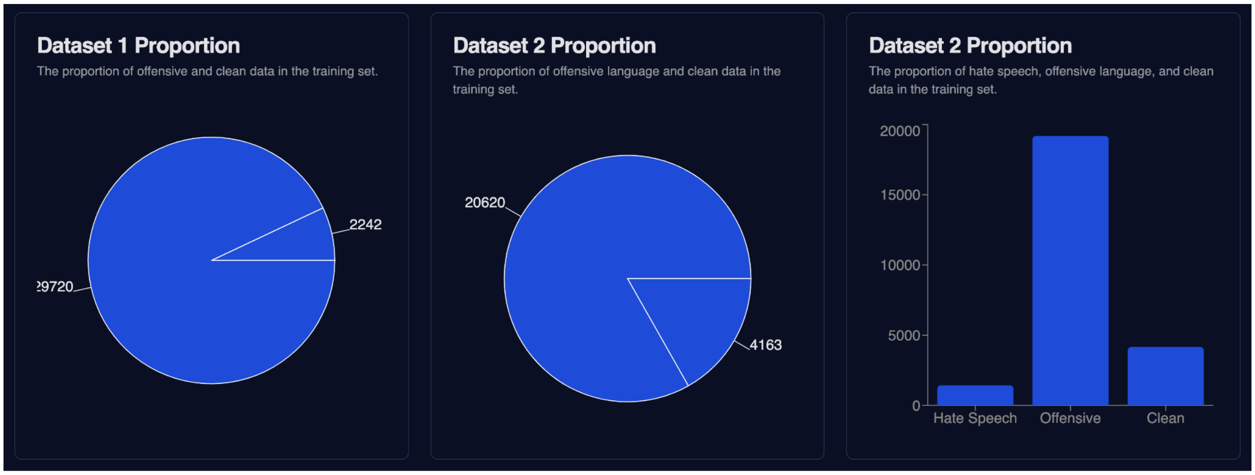
Total Dataset Proportion

The proportion of offensive language and clean data combined in both training sets.

Word Cloud

The most common words in the dataset

Visual 2: Charts about the data



Dataset 1 Proportion

The proportion of offensive and clean data in the training set.

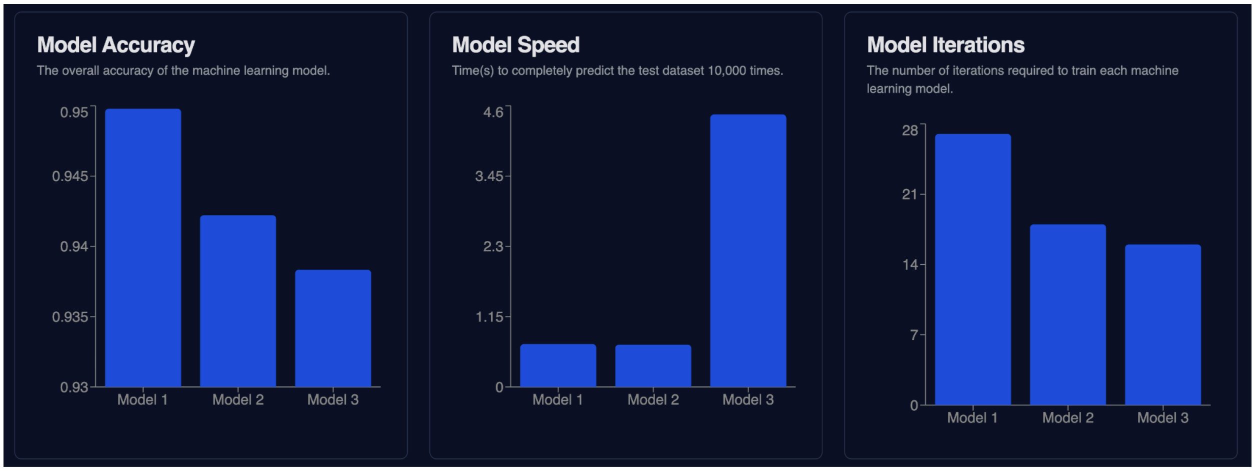
Dataset 2 Proportion

The proportion of offensive language and clean data in the training set.

Dataset 2 Proportion

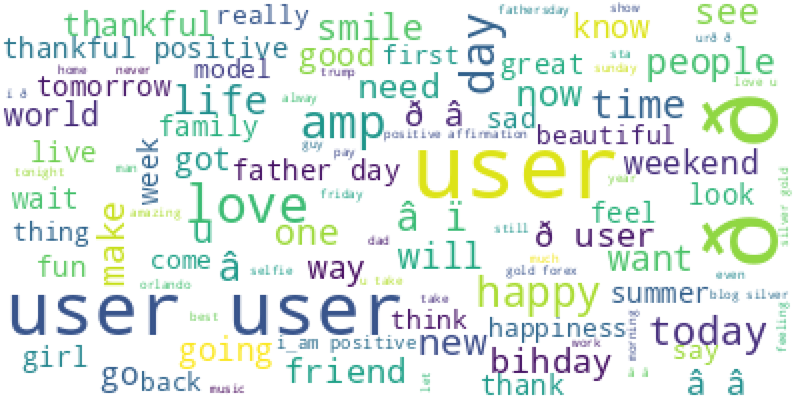
The proportion of hate speech, offensive language, and clean data in the training set.

Visual 3: Charts about the model

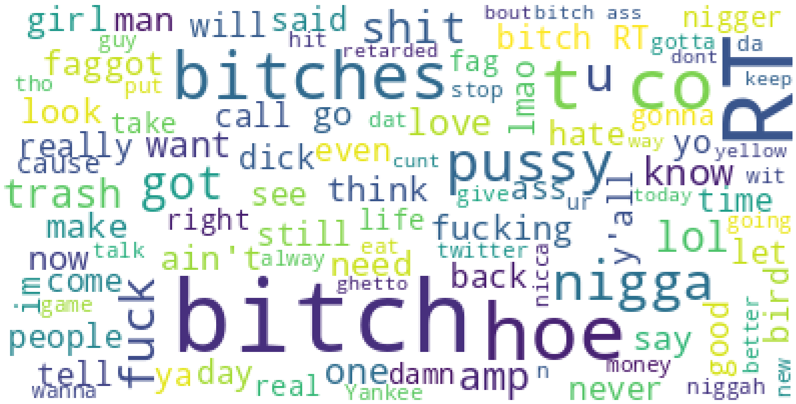


Visualizations 4-6 are wordcloud images sourced from the development process and showcase the most prominent words sourced from the datasets.

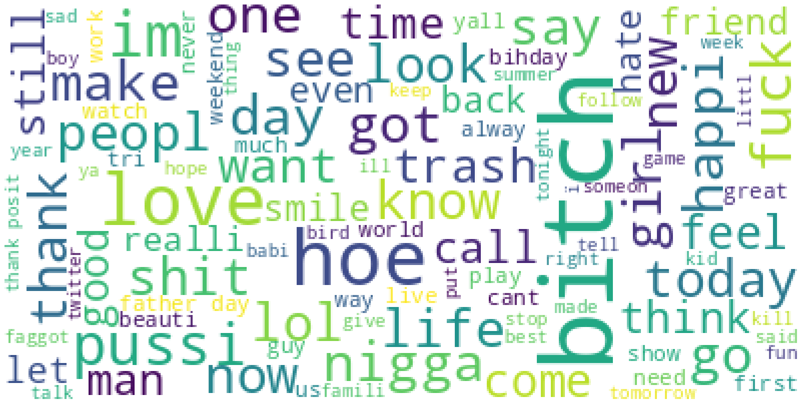
Visual 4: Wordcloud of Training Dataset 1



Visual 5: Wordcloud of Training Dataset 2

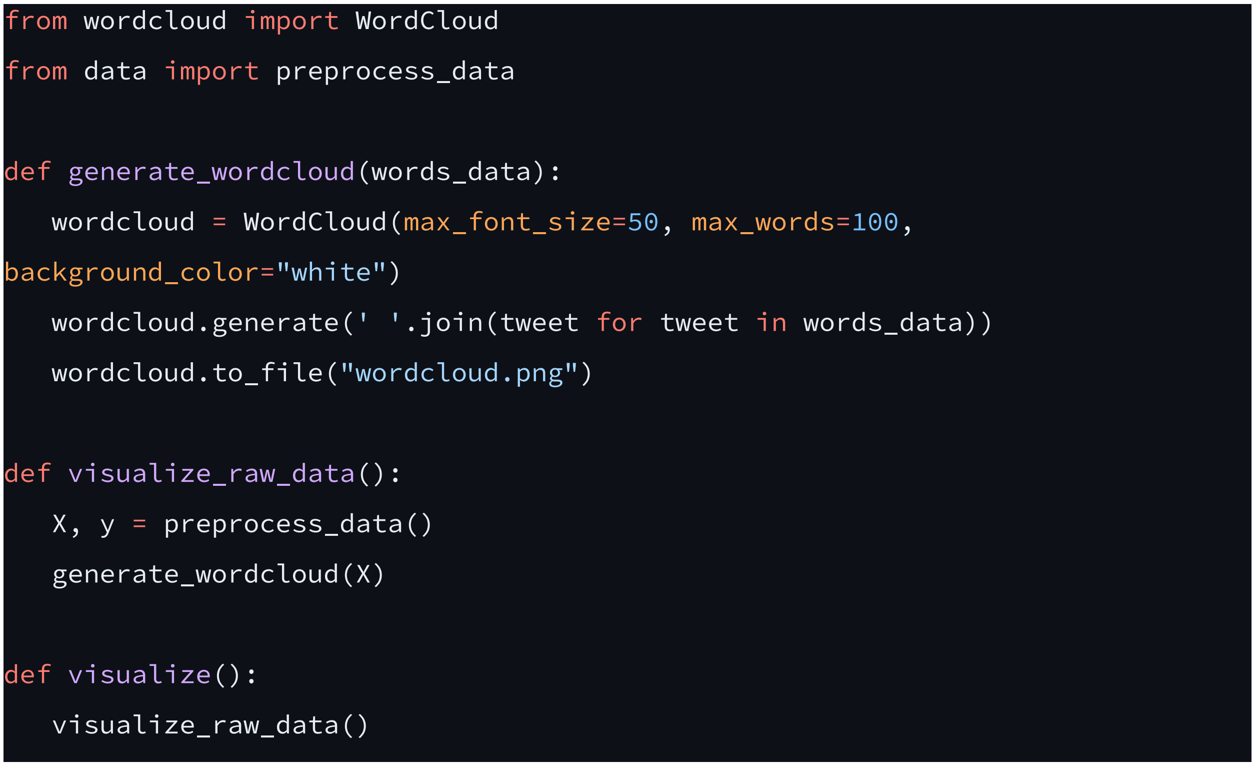


Visual 6: Wordcloud of Training Dataset 1 & 2 with extensive cleaning



The following is the code used to generate the wordcloud image for the data used to train Model 3 (Visual 6). The other wordcloud images were generated using similar code.

visualization.py



```
from wordcloud import WordCloud
from data import preprocess_data

def generate_wordcloud(words_data):
   wordcloud = WordCloud(max_font_size=50, max_words=100,
background_color="white")
   wordcloud.generate(' '.join(tweet for tweet in words_data))
   wordcloud.to_file("wordcloud.png")

def visualize_raw_data():
   X, y = preprocess_data()
   generate_wordcloud(X)

def visualize():
   visualize_raw_data()
```

User Guide

The following user guide provides instructions on how to access and use the application either via the hosted web application or via self-hosting the development environment.

Accessing the website

To access the website, follow the steps below:

1. Open a web browser on your desktop or mobile device. The site is built to be responsive and should work on most devices but works best on desktops.
2. Enter the URL of the website (wgu-capstone-xavier-loera-flores.vercel.app) in the address bar and press enter to navigate to the website.
3. Modules of the site may take a few seconds to load as they are set to sleep during periods with low activity. Once the website loads, you will be presented with the homepage.
4. Once the website loads, you will be presented with the homepage.

Navigating the Website

Once you are on the home page, you can navigate the website using the following steps:

1. Use the navigation menu at the top of the page to access different sections of the website.
2. Click on the links to navigate to specific pages or click on the toggle buttons to perform actions.
3. The following pages are available on the website:
    - Home: The main landing page of the website where users can view the timeline or make a post.
    - About: View supplemental information about the site and its creation.
    - Analysis: View data analysis or visualizations related to the data or the machine learning solution.
    - Docs: Access documentation for the api.
4. The following actions can be performed via the toggle buttons on the nav bar.
    - Dark Mode: Toggle between light and dark mode.
    - Content Filter: Filter the content on the page based for offensive content.

Using the Main Application

To use the application, follow these steps:

1. On the homepage, you can scroll and view posts that other users have made. You can navigate between pages of posts using the navigation buttons at the bottom of the page.
2. Using the text box at the top of the page, you can compose a post by entering your text.
3. You now have the option to either submit your post or to simply see how the model would classify your post.
4. After submitting to classifying a post, you will see notifications about your post pop up on the bottom right of the screen.

Viewing Analysis

To view analysis, follow these steps:

1. Navigate to the Analysis tab using the navigation bar and select a specific analysis page to view.
2. You can view visualizations and analysis of the data or machine learning model on the analysis page.
3. You can also download raw files relating to the page using the dropdown menu near the top of the page.

Accessing Documentation and Other Information

There are other pages on the site which are purely informational. You can access these pages by clicking on the links in the navigation bar.

1. To access the API documentation, click on the Docs link in the navigation bar.
2. To access information about the site and its creation, click on the About tab in the navigation bar and select the specific About page you would like to view.

Running the Application Locally

Alternatively, you can run modules of the entire program locally using the instructions in the README.md file for each of the modules directly.
Once you have downloaded the entire repository, you will see the following directories:

- backend
- frontend
- data
- database

- model
- project_management
- docs

You can run the backend, frontend, database, model, and data modules locally by following the instructions in the README.md file in each of the directories. The only modules needed to run the application are the backend, database, and frontend modules. The other modules are used for data processing, model training, project management, and documentation. Once you have the database, backend, and frontend modules running, you should be able to access the site locally by navigating to the localhost address provided by the frontend module.

References

(1) Hotz, Nick "What is CRISP DM?" Data Science Process Alliance https://www.datascience-pm.com/crisp-dm-2/ Accessed Feb 20,2024.

(2) Toosi, Ali "Twitter Sentinment Analysis" Kaggle https://www.kaggle.com/datasets/arkhoshghalb/twitter-sentiment-analysis-hatred-speech Accessed Feb 20,2024.

(3) Samoshyn, Andrii "Hate Speech & Offensive Language" Kaggle https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset/data Accessed Feb 20,2024.