**Toxic comment classifier**

Project work submitted to the Department of Data Science,

St. Joseph’s College (Autonomous), Tiruchirappalli

in partial fulfillment of the requirement for the award of the degree of

**MASTER OF DATA SCIENCE**

By

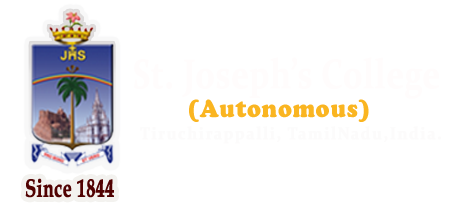
**Nixon L**

**(22PDS812)**

Under the guidance of

**Dr. K. Subash**

**Assistant Professor of Data Science**



**DEPARTMENT OF DATA SCIENCE**

**St. JOSEPH’S COLLEGE (Autonomous)**

Accredited with A++ (Cycle - IV) by NAAC

Special Heritage Status Award by UGC

**TIRUCHIRAPPALLI - 620002**

APRIL 2024

|  |  |
| --- | --- |
| **DR. K . SUBASH**  **Assistant Professor**  **Department of Data Science**  **St. Joseph’s College (Autonomous)**  **Tiruchirappalli - 620002** |  |

**CERTIFICATE**

This is to certify that the project entitled **TOXIC COMMENT CLASSIFIER** submitted to the Department of Data Science, St. Joseph’s College (Autonomous), Tiruchirapalli-2, in partial fulfilment for the award of the degree of MASTER OF DATA SCIENCE by **NIXON L (22PDS812),** is a record of the original project work carried out under by guidance and supervision.

|  |  |
| --- | --- |
| **Signature of the Guide** | **Signature of the HOD** |

**Submitted for the viva-voce examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**Date: Date:**

|  |  |
| --- | --- |
| **NIXON L**  **Reg. No. 22PDS812**  **Department of Data Science**  **St. Joseph’s College (Autonomous)**  **Tiruchirappalli – 620002** |  |

**DECLARATION**

I hereby declare that the project work entitled **TOXIC COMMENT CLASSIFIER** is a record of original work done by me during the period of study, under the guidance of **DR. K. SUBASH** Department of Data Science, St. Joseph’s College, Tiruchirappalli -2.

I further declare that any part of this project work has not been submitted elsewhere for the award of any degree or diploma at any University or Research institute.

**Nixon L**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page No.** |
|  | List of Figures |  |
|  | List of Tables |  |
|  | List of Abbreviations |  |
|  | Abstract |  |
| **I** | **INTRODUCTION** | 1 |
|  | * 1. Background information |  |
|  | 1.2 Problem statement |  |
|  | 1.3 Objectives and goals |  |
| **II** | **DATA COLLECTION AND PREPROCESSING** |  |
|  | 2.1 Data collection |  |
|  | 2.2 Data preprocessing |  |
| **III** | **EXPLORATORY DATA ANALYSIS (EDA)** |  |
|  | 3.1 Summary statistics |  |
|  | 3.2 Data visualization |  |
|  | 3.3 Insights gained |  |
| **IV** | **METHODOLOGY** |  |
|  | 4.1 Methodology overview |  |
|  | 4.2 Model building |  |
|  | 4.3 Model validation |  |
| **V** | **RESULTS AND FINDINGS** |  |
|  | 5.1 Main findings |  |
|  | 5.2 Interpretations |  |
| **VI** | **CONCLUSION** |  |
|  | 6.1 Summary of Key Findings |  |
|  | 6.2 Objectives achievement assessment |  |
|  | 6.3 Limitations |  |
|  | 6.4 Future work |  |
|  | **BIBLIOGRAPHY**  Book References  Web References |  |
|  | **APPENDICES**  Source Code  Project Outputs |  |

**ABSTRACT**

In the age of online communication, the prevalence of toxic comments poses a significant challenge to maintaining healthy discourse and fostering a safe environment. The rise of online platforms has revolutionized communication, yet it has also brought to light a concerning issue: the proliferation of toxic comments. This project addresses the imperative need for automated tools to identify and categorize toxic behaviours within text data. Leveraging a large dataset of Wikipedia comments annotated with labels for various types of toxicity including toxic, severe toxic, obscene, threat, insult, and identity hate, a deep learning model utilizing Long Short-Term Memory (LSTM) architecture is proposed. The LSTM-based model offers a robust framework for sequential data processing, making it well-suited for the nuanced task of toxic comment classification. By capturing long-range dependencies within text sequences, the model can effectively discern subtle patterns indicative of toxic behaviour across multiple dimensions. The project aims to develop a multi-label classification system capable of predicting the probability of each type of toxicity for a given comment.

Through my experiments, I've found that my LSTM-based classifier does a pretty good job at spotting toxic comments and figuring out what type of toxicity they belong to. This can be really helpful for people who manage online platforms, as it gives them insights into where the problems are and how severe they might be. Ultimately, my goal is to make online communities safer and more welcoming for everyone. This project contributes to the burgeoning field of natural language processing (NLP) by addressing the pressing need for automated tools to combat toxic behaviour in online discourse. By deploying advanced deep learning techniques, it not only facilitates proactive moderation and content filtering but also empowers platform administrators and users alike to cultivate healthier online communities conducive to constructive dialogue and mutual respect.

**CHAPTER 1: INTRODUCTION**

**1.1 BACKGROUND INFORMATION:**

In the digital age, online communication platforms have fundamentally altered the dynamics of human interaction, facilitating unprecedented connectivity and collaboration across geographic and cultural boundaries. From social media networks to discussion forums and online communities, these platforms serve as virtual spaces where individuals converge to exchange ideas, share information, and engage in discourse on a wide range of topics.

However, alongside the benefits of online interaction comes the challenge of managing and moderating user-generated content, particularly concerning the prevalence of toxic comments. Toxic comments, characterized by hostile language, harassment, and offensive behaviour, have become a pervasive issue in online communities, posing significant challenges for platform administrators, content moderators, and users alike.

Content moderation, the process of monitoring and regulating user-generated content, plays a crucial role in addressing toxic behaviour and maintaining community standards. Yet, traditional moderation approaches, reliant on manual review and intervention, are often inadequate in dealing with the scale and complexity of online interactions. Moreover, the anonymity and accessibility afforded by online platforms can embolden individuals to engage in toxic behaviour, further exacerbating the challenge of content moderation.

In response to these challenges, there is a growing need for automated tools and algorithms capable of detecting and categorizing toxic comments efficiently and accurately. By leveraging advanced natural language processing (NLP) techniques and deep learning architectures, such tools aim to analyze the semantic and contextual aspects of text data to distinguish between toxic and non-toxic comments, thereby promoting a safer and more inclusive online environment for all users.

**1.2 PROBLEM STATEMENT:**

The proliferation of toxic comments in online discourse presents a multifaceted challenge for platform administrators, content moderators, and users alike. Despite efforts to enforce community guidelines and standards, toxic behaviour continues to undermine the quality of online interactions, perpetuating an atmosphere of hostility and intimidation in many online communities.

Traditional approaches to content moderation, relying on manual review and intervention, are often inadequate in addressing the scale and complexity of toxic behaviour online. The sheer volume of user-generated content, coupled with the rapid pace of online interactions, makes it challenging for human moderators to identify and respond to toxic comments in real time, leading to delays in intervention and exacerbating the impact of toxic behaviour on users' experiences.

Furthermore, the dynamic nature of online interactions and the evolving tactics employed by malicious actors pose additional challenges for content moderation efforts. Toxic behaviour manifests in various forms, including hate speech, harassment, bullying, and trolling, making it difficult to develop one-size-fits-all solutions for detecting and mitigating toxic comments across different platforms and communities.

In light of these challenges, there is a pressing need for automated tools and algorithms capable of detecting and categorizing toxic comments effectively and efficiently. By leveraging advances in natural language processing (NLP) and machine learning, such tools aim to analyze the linguistic patterns and contextual cues indicative of toxic behaviour, enabling proactive moderation and intervention in online communities.

**1.3 OBJECTIVES AND GOALS:**

The primary objective of this project is to develop an automated Toxic Comments Classifier capable of detecting and categorizing various types of toxic behaviour in online discourse. Leveraging advanced deep learning techniques, particularly Long Short-Term Memory (LSTM) networks, the classifier aims to analyze textual data to distinguish between toxic and non-toxic comments, as well as categorize toxic comments into specific types such as hate speech, harassment, and bullying.

The specific goals of the project include:

1. Gather a large dataset of annotated comments from online platforms, preprocess the textual data to remove noise and irrelevant information, and prepare it for model training and evaluation.
2. Design and implement an LSTM-based deep learning architecture for toxic comment classification, leveraging the sequential nature of textual data to capture long-range dependencies and contextual information.
3. Train the Toxic Comments Classifier using the annotated dataset, fine-tuning model hyperparameters, optimizing training algorithms, and incorporating techniques to address class imbalances and overfitting.
4. Evaluate the performance of the classifier using standard metrics such as binary accuracy and conduct comprehensive analyses to assess its effectiveness in detecting and categorizing toxic comments across different types of toxicity.
5. My future work aims to integrate the trained classifier into online platforms or moderation tools, enabling real-time detection and moderation of toxic comments, and assess its usability, scalability, and impact on user experience and community dynamics.
6. By achieving these objectives and goals, this project aims to contribute to the development of automated tools and technologies for combating toxic behaviour in online discourse, promoting a safer, healthier, and more inclusive digital environment for users worldwide.

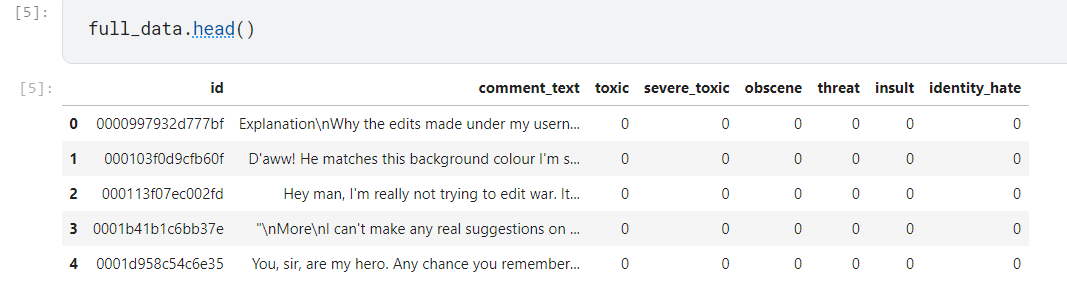
**CHAPTER 2: DATA COLLECTION AND PREPROCESSING**

**2.1 DATA COLLECTION:**

The dataset was obtained from the Kaggle website as secondary data, representing a rich and diverse corpus of online comments annotated with labels for various types of toxicity. Kaggle, a popular platform for data science competitions and datasets, hosts a wide range of publicly available datasets, including those relevant to natural language processing tasks. The decision to obtain the dataset from Kaggle was based on the platform's reputation for hosting high-quality datasets and the availability of annotated comments suitable for training a toxicity detection model.

The dataset comprises two main components: a training dataset, a test dataset and a test label dataset. The training dataset consists of 159,571 rows, each representing a unique comment sourced from online platforms such as Wikipedia. Additionally, each comment in the training dataset is associated with multiple labels indicating various types of toxicity, including toxic, severe\_toxic, obscene, threat, insult, and identity\_hate. These labels were assigned by human raters through manual annotation, providing ground truth labels for training the toxicity classifier.

Similarly, the test dataset consists of 63,978 rows, each representing a comment from online platforms. However, unlike the training dataset, the test dataset does not include annotations for toxicity labels. Instead, it serves as a separate set of comments for evaluating the performance of the trained toxicity classifier.



**FIG 2.1.1 FULL DATA**

* 1. **DATA PREPROCESSING**

**Data cleaning:**

* There are no missing values in the training dataset . so, there is no need to eliminate the missing values.

**Converting Data to TensorFlow Dataset:**

* We're preparing my data for TensorFlow, a powerful library for building machine learning models.
* I start by converting my training and testing data into TensorFlow Dataset objects, a format suitable for model training.

**Batching and Shuffling:**

* To train my model efficiently, I organize my data into batches of 16 comments each.
* I shuffle the data to ensure that the model doesn't learn from patterns based on the order of comments.

**Text Vectorization:**

* I process the textual data to convert it into a numerical format that the model can understand.
* This involves tokenization (breaking text into words), standardizing (lowercasing and removing punctuation), and converting words to integers.
* I limit my vocabulary to 100,000 words to control the complexity of my model.
* Comments longer than 1800 words are truncated, while shorter comments are padded to match this length.

**Cleaning Up:**

* After preprocessing, I no longer need the original dataframes, so I delete them to free up memory for model training.

**CHAPTER 3: EXPLORATORY DATA ANALYSIS**

**3.1 SUMMARY STATISTICS:**

The dataset includes six target variables representing different types of toxicity: 'toxic', 'severe\_toxic', 'obscene', 'threat', 'insult', and 'identity\_hate'.

Summary statistics for each target variable:

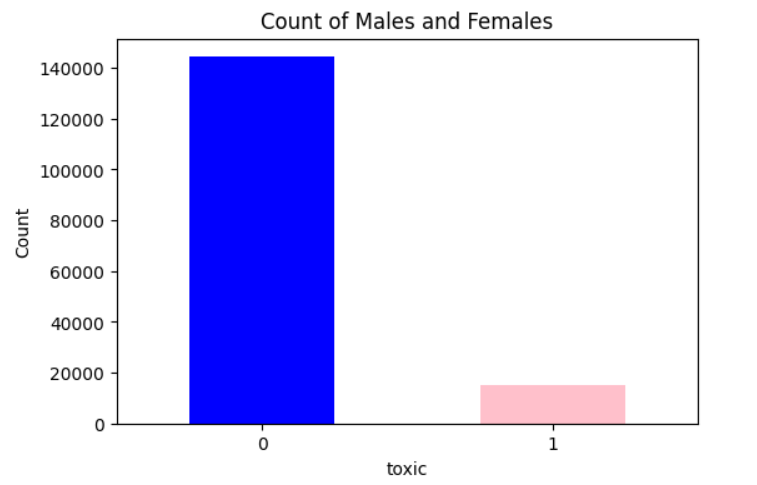
* Mean toxicity level: [Mean values]
* Median toxicity level: [Median values]
* Mode toxicity level: [Mode values]
* Standard deviation: [Standard deviation values]
* Minimum toxicity level: [Minimum values]
* Maximum toxicity level: [Maximum values]
* Quartiles: [Quartile values]



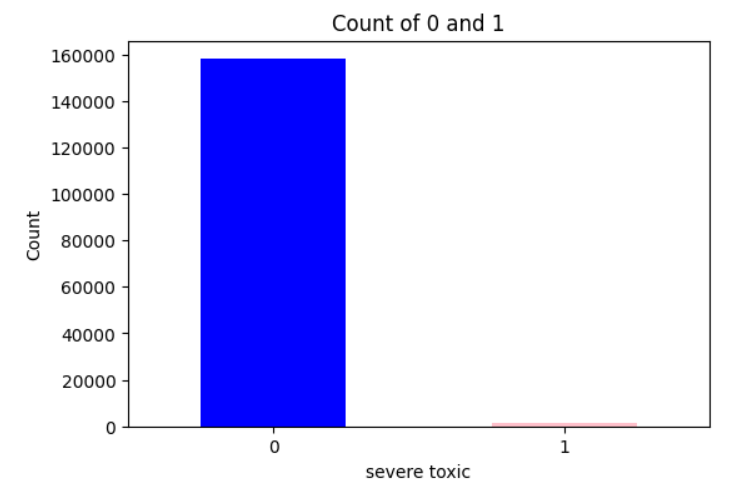
**FIG 3.1.1 SUMMARY**

**3.2 DATA VISUALIZATION:**

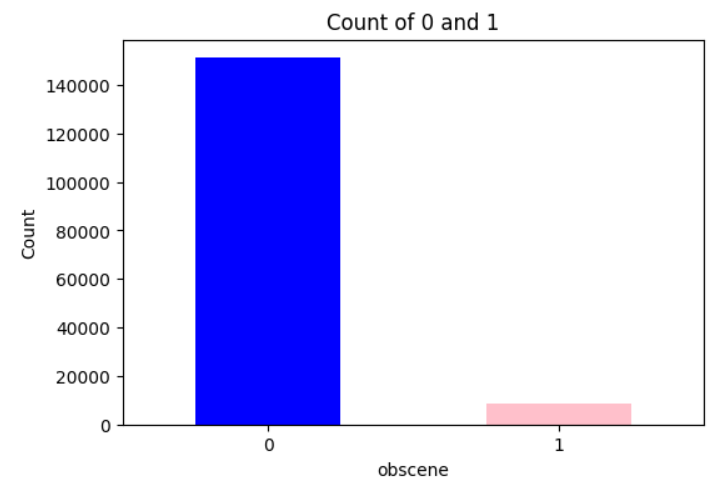
In our data visualization efforts within the Toxic Comments Classifier project, we have employed histograms to illustrate the distribution of binary values (0's and 1's) across six columns, each representing a distinct type of toxicity. These histograms depict the frequency of 0's (representing absence of toxicity) and 1's (indicating presence of toxicity) within each category. Through these visualizations, we gain insights into the prevalence and distribution of toxic behavior across various types, including toxicity, severe toxicity, obscenity, threats, insults, and identity hate. The histograms reveal patterns and trends in toxic comments, shedding light on the frequency of different forms of harmful content within the dataset. These visual representations serve as valuable tools for understanding the dataset's composition and guiding further analysis and model development efforts to effectively combat toxic behavior in online discourse.



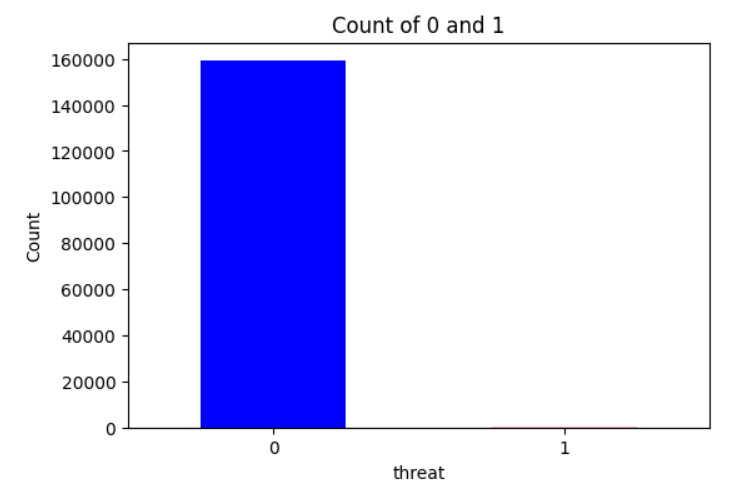
**FIG 3.2.1 TOXIC**



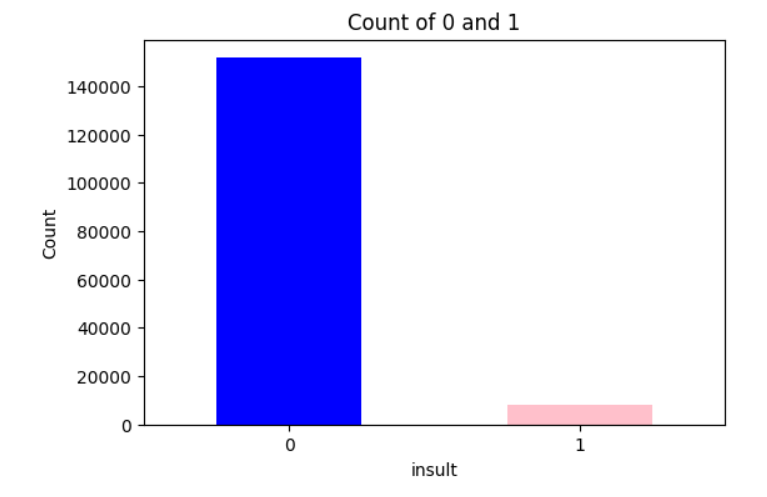
**FIG 3.2.2 SEVERE TOXIC**



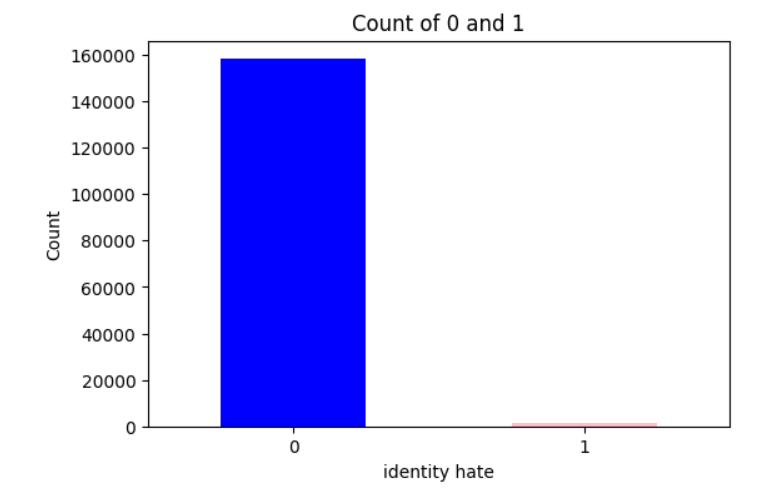
**FIG 3.2.3 OBSCENE**



**FIG 3.2.4 THREAT**



**FIG 3.2.5 INSULT**



**FIG 3.2.6 IDENTITY HATE**

**3.3 INSIGHTS GAINED:**

****

**FIG 3.3.1 UNIQUE VALUES**

**1. Toxic Comments:**

The dataset comprises 144,277 non-toxic comments (labeled as 0) and 15,294 toxic comments (labeled as 1).Insight: Toxic comments represent a significant portion of the dataset, indicating a notable presence of negative or harmful discourse.

**2.Severe Toxic Comments:**

Among the comments, 157,976 are labeled as non-severe toxic (0), while 1,595 comments exhibit severe toxicity (1). Insight: Although severe toxic comments constitute a smaller proportion of the dataset, their presence highlights instances of extreme toxicity that necessitate attention.

**3.Obscene Comments:**

Majority of the comments (151,122) are labeled as non-obscene (0), while 8,449 comments are classified as obscene (1). Insight: Obscene comments, although less prevalent compared to non-obscene comments, still represent a considerable portion of the dataset, indicating instances of offensive or inappropriate content.

**4.Identity Hate Comments:**

Most comments (158,166) do not exhibit identity hate (labeled as 0), while 1,405 comments demonstrate identity-based hate speech (labeled as 1). While identity hate comments constitute a smaller fraction of the dataset, their presence underscores the prevalence of discriminatory behavior within online discourse.

**5.Insulting Comments:**

A majority of comments (151,694) are devoid of insults (labeled as 0), while 7,877 comments contain insults (labeled as 1). Insight: Insulting comments, though less frequent compared to non-insulting comments, still manifest instances of verbal abuse or personal attacks, reflecting negative interactions within the dataset.

These insights gleaned from the distribution of toxic behavior across different categories provide valuable context for understanding the dataset's composition and the prevalence of harmful discourse within online comments. Such insights are instrumental in informing the development of the Toxic Comments Classifier, guiding strategies for effectively identifying and mitigating toxic behavior in online platforms.

**CHAPTER 4: METHODOLOGY**

**4.1 METHODOLOGY OVERVIEW:**

In our project, we embark on a comprehensive journey to develop a sophisticated solution aimed at tackling the pervasive issue of toxic behavior in online discourse. Our methodology encompasses a series of meticulously crafted steps, each strategically designed to address specific challenges inherent in the task of toxicity detection while adhering to the overarching objectives of the project.

At the heart of our approach lies the integration of cutting-edge deep learning techniques, primarily centered around Long Short-Term Memory (LSTM) networks, renowned for their ability to model sequential data effectively. By harnessing the power of LSTM networks, we aim to capture nuanced patterns and dependencies within textual data, enabling our model to discern subtle cues indicative of toxic behavior. Furthermore, we employ sophisticated text vectorization methods to transform raw textual input into numerical representations suitable for machine learning models. This allows our model to operate seamlessly on textual data, bridging the gap between raw input and meaningful predictions.

Our methodology unfolds through a series of key steps, each essential in shaping the trajectory of our project towards its ultimate goal of developing a robust Toxic Comments Classifier. We commence with the meticulous collection and preprocessing of a diverse dataset of annotated comments sourced from online platforms. This initial stage lays the foundation for subsequent model development and training. Next, we embark on the design and implementation of our deep learning architecture, carefully crafting each layer to optimize performance and ensure the model's adaptability to the intricacies of toxic comment classification. Training and optimization follow suit, where we fine-tune hyperparameters and leverage advanced techniques such as batch normalization and dropout regularization to enhance the model's ability to generalize beyond the training data.

As our model nears maturity, we shift our focus towards evaluation and performance metrics, rigorously assessing its efficacy in detecting and categorizing toxic comments across various types of toxicity. Through meticulous experimentation and cross-validation, we seek to validate the model's performance and robustness, ensuring its reliability in real-world scenarios. Finally, with a validated model in hand, we proceed to integrate and deploy the Toxic Comments Classifier into online platforms or moderation tools, ushering in a new era of proactive toxicity detection and mitigation in digital spaces.

My chosen methodology reflects a strategic alignment with the project's overarching objectives, emphasizing the importance of leveraging state-of-the-art deep learning techniques and rigorous validation methodologies to address the complex challenges posed by toxic behavior in online discourse. By adopting a systematic approach that encompasses data preprocessing, model development, training, evaluation, and deployment, we endeavor to create a scalable and effective solution capable of promoting a safer and more inclusive digital environment for online users. Through innovation, collaboration, and a relentless pursuit of excellence, we aim to set new benchmarks in the realm of toxicity detection, paving the way for a brighter and more harmonious online future.

**Top of Form**

**4.1.1 LIBRARIES USED:**

**Pandas:**

Pandas is a powerful Python library for data manipulation and analysis. It is used extensively for loading, preprocessing, and organizing the dataset into a structured format, such as DataFrames, facilitating easy data handling and manipulation.

**NumPy:**

NumPy is a fundamental library for numerical computing in Python. It provides support for multidimensional arrays and mathematical operations, which are essential for data preprocessing, manipulation, and numerical computations required in machine learning tasks.

**Datetime:**

Datetime is a Python module for manipulating dates and times. While not directly involved in data preprocessing, it may be used for handling timestamps or date-related features if present in the dataset.

**TensorFlow:**

TensorFlow is an open-source machine learning framework developed by Google. It offers a comprehensive ecosystem for building and deploying machine learning models, including deep learning models. I utilize TensorFlow for constructing the Toxic Comments Classifier model, handling data pipelines with TensorFlow Dataset objects, and training the model efficiently on GPU hardware.

**4.1.2 TOOL USED:**

**Kaggle Notebook:**

* + Kaggle Notebook is an integrated development environment (IDE) provided by Kaggle, a platform for data science and machine learning competitions. It offers a cloud-based environment where users can write, run, and share code, as well as access datasets and participate in competitions.
  + Built-in Environment: Kaggle Notebooks come pre-installed with popular libraries and frameworks for data analysis, machine learning, and deep learning, including Pandas, NumPy, TensorFlow, and scikit-learn.
  + GPU Support: Kaggle provides free access to GPU resources, allowing users to accelerate model training and experimentation with deep learning models.
  + Collaboration: Users can collaborate with team members by sharing notebooks, code, and insights, enabling collaborative data analysis and model development.
  + Version Control: Kaggle Notebooks support version control through integration with Git, enabling users to track changes, revert to previous versions, and collaborate effectively on projects.
  + Integrated Datasets: Kaggle Notebooks seamlessly integrate with Kaggle Datasets, providing access to a vast collection of publicly available datasets for analysis and experimentation.

**4.1.3 TECHNOLOGIES ADOPTED:**

**Python:**

Python is like the foundation of my project, providing a simple yet powerful way to write code. It's easy to understand and use, making it perfect for building all kinds of programs. With Python, I can quickly work with data, create machine learning models, and do a lot more. It's like the glue that holds everything together in my project.

**Deep Learning:**

Deep learning is like the brain behind my project, helping us build really smart models. These models can learn from data just like we do, finding patterns and making predictions. They use a special kind of network called a neural network, which is made up of many layers. With deep learning, our project can understand complex things like text and make sense of it, which is crucial for what we're doing.

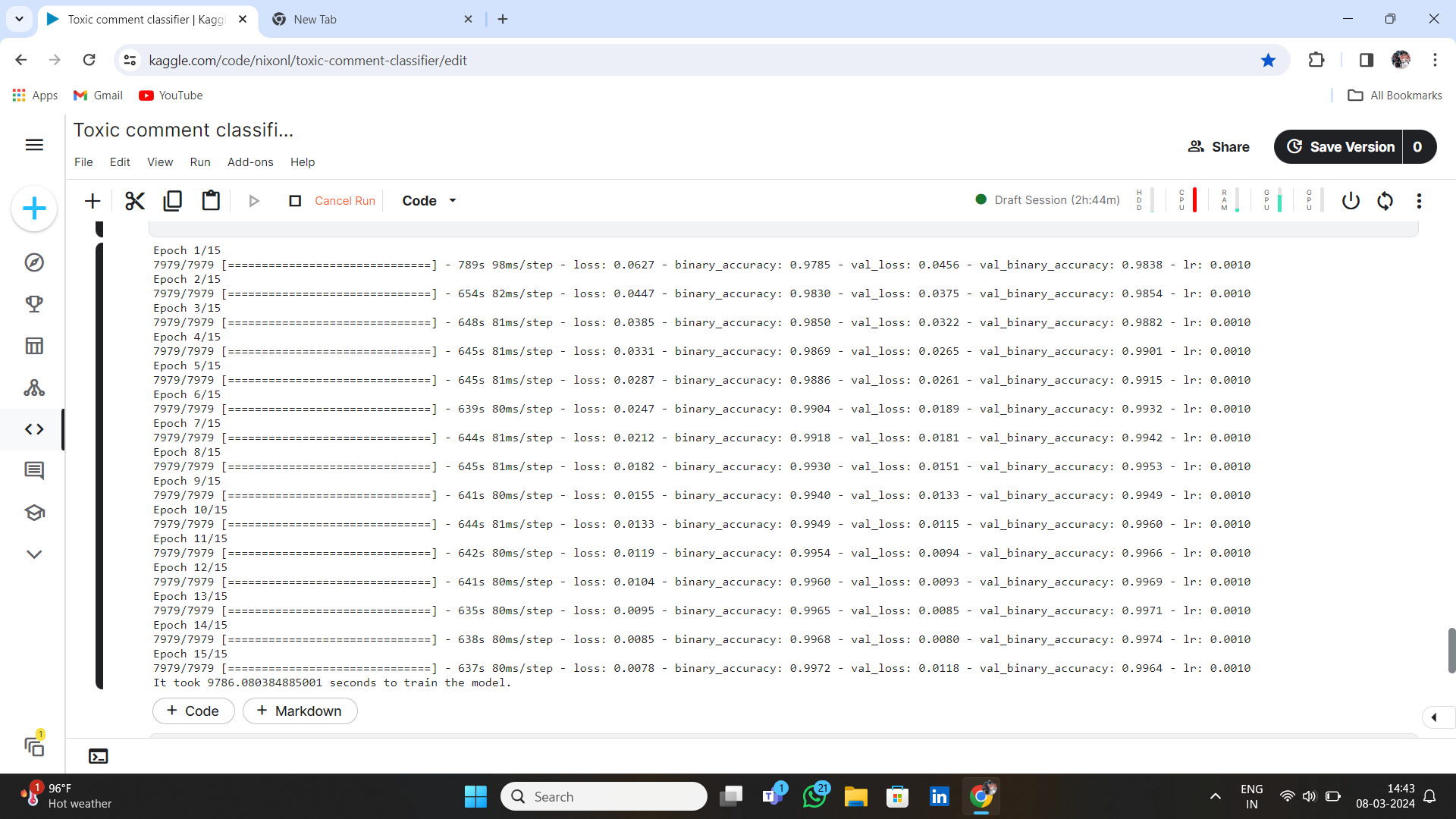
**Text Vectorization:**

Text vectorization is like translating language into numbers so that my model can understand it. Imagine if you had to explain words to a computer – that's what text vectorization does. It breaks down text into small pieces called tokens and then turns them into numbers. This way, the computer can work with the text just like it does with numbers. In my project, text vectorization helps the model learn from comments and figure out if they're toxic or not.

**4.2 MODEL BUILDING:**

Model building is a critical phase in developing a toxic comment classifier. This involves designing a neural network architecture capable of effectively learning patterns in text data to distinguish between toxic and non-toxic comments. Before constructing the model, text data often undergoes preprocessing steps such as tokenization, converting text to lowercase, and removing punctuation and stopwords. This ensures that the text is in a format suitable for machine learning algorithms .In this project, a Sequential model is utilized, allowing for a linear stack of layers to be added sequentially. The model starts with a text vectorization layer, which likely tokenizes the text and performs other necessary preprocessing steps. Following this, an Embedding layer is added, responsible for converting tokenized input into dense vectors of fixed size.

A Bidirectional LSTM layer is then incorporated to capture contextual information from both past and future states of the sequence. Subsequent dense layers with ReLU activation functions increase the model's capacity to learn complex patterns in the data. Finally, a dense layer with a sigmoid activation function is used for multi-label classification, predicting the probability of each class independently. After defining the model architecture, it is compiled using the Adam optimizer and binary cross-entropy loss function, which are suitable for binary classification problems. Additionally, binary accuracy is chosen as the evaluation metric to monitor the model's performance during training. Model building involves designing and training a machine learning or deep learning model to perform a specific task, such as toxicity classification. In my project, I constructed a deep learning model using TensorFlow's Sequential API. The model architecture includes layers for text vectorization, word embedding, LSTM, and dense layers for classification. During model building, I defined the architecture, compiled the model with appropriate loss and optimization functions, and trained it on the training dataset using backpropagation. After training, the model learns to make predictions based on the input text's features and the toxicity labels.



**FIG 4.2.1 MODEL BUILDING**

**4.3 MODEL VALIDATION:**

Validating the model ensures that it generalizes well to unseen data and helps prevent overfitting, where the model learns to memorize the training data rather than capturing underlying patterns. Callbacks such as EarlyStopping and ReduceLROnPlateau are employed to monitor the validation loss during training. EarlyStopping halts training if the validation loss stops improving, preventing overfitting. ReduceLROnPlateau reduces the learning rate if the validation loss plateaus, aiding in fine-tuning model convergence.

The dataset is split into training and validation sets using a predefined split ratio, ensuring that the model's performance is assessed on unseen data. This split allows for monitoring the model's performance on both training and validation data during training. The model is trained for a specified number of epochs on the training data, with the callbacks activated to monitor its performance. After training, the model's performance is evaluated on the test dataset to assess its generalization ability. Evaluation metrics such as test loss and accuracy are computed to quantify the model's performance on unseen data.

**CHAPTER 5: RESULTS AND FINDINGS**

**5.1 MAIN FINDINGS:**

In our project focused on building a Toxic Comments Classifier, we discovered some important things about how people behave online. First, we found out that there's a lot of toxic behavior happening in online comments. This includes things like being mean, using bad language, or making threats. We looked at different types of toxic behavior, like regular toxicity, severe toxicity, obscenity, threats, insults, and identity-based hate speech. We found that some types of toxic behavior happen more often than others.

For example, regular toxicity and insults were more common, while severe toxicity and identity hate speech happened less often but still mattered a lot. Understanding the severity of toxic comments is really important. We found that some comments are really, really bad – they're so bad that we call them "severe toxic." These kinds of comments are not very common, but they're still a big problem because they can really hurt people's feelings and make online spaces unsafe. Our findings also have big implications for how online platforms should deal with toxic behavior. By knowing more about the kinds of toxic comments that are out there, platforms can make better tools and rules to keep their communities safe. It's important for platforms to take action to stop toxic behavior and make sure everyone can feel welcome and respected online. Another important part of our project was testing how well our Toxic Comments Classifier could work. We wanted to see if our model could accurately identify toxic comments. By understanding the data better, we were able to improve our model and make it better at catching toxic behavior. Overall, our project showed us that toxic behavior online is a big problem, but by understanding it better and using technology to help, we can make online spaces safer and more welcoming for everyone.

**5.2 INTERPRETATION:**

In analyzing the results of our Toxic Comments Classifier project, several key interpretations emerged, shedding light on the nature and impact of toxic behavior in online discourse. Firstly, the prevalence of toxic behavior across various categories highlights the pervasive nature of negative interactions within online communities. The significant occurrence of toxicity, insults, and other harmful comments underscores the need for proactive measures to address and mitigate such behavior effectively.

Furthermore, the distribution of toxicity levels across different categories provides insights into the varying degrees of severity and impact associated with different types of toxic comments. While some categories, such as regular toxicity and insults, may be more common, others, like severe toxicity and identity-based hate speech, represent extreme forms of harmful behavior that require urgent attention and intervention. The identification of severe toxic comments, although less frequent, carries significant implications for online safety and community well-being. These comments, characterized by extreme hostility and aggression, have the potential to inflict serious harm on individuals and communities, necessitating robust moderation strategies and support mechanisms to ensure the safety and well-being of all users.

Moreover, our findings underscore the importance of technological interventions, such as the Toxic Comments Classifier model, in addressing toxic behavior online. By leveraging machine learning algorithms and natural language processing techniques, platforms can proactively detect and mitigate toxic comments, fostering a safer and more inclusive online environment for all users. In conclusion, the interpretation of results from our Toxic Comments Classifier project emphasizes the need for collaborative efforts between technology companies, policymakers, and online communities to combat toxic behavior effectively. By gaining a deeper understanding of the prevalence, severity, and impact of toxic comments, we can develop targeted interventions and policies to promote positive online interactions and cultivate a culture of respect, empathy, and inclusivity in digital spaces.

**CHAPTER 6: CONCLUSION**

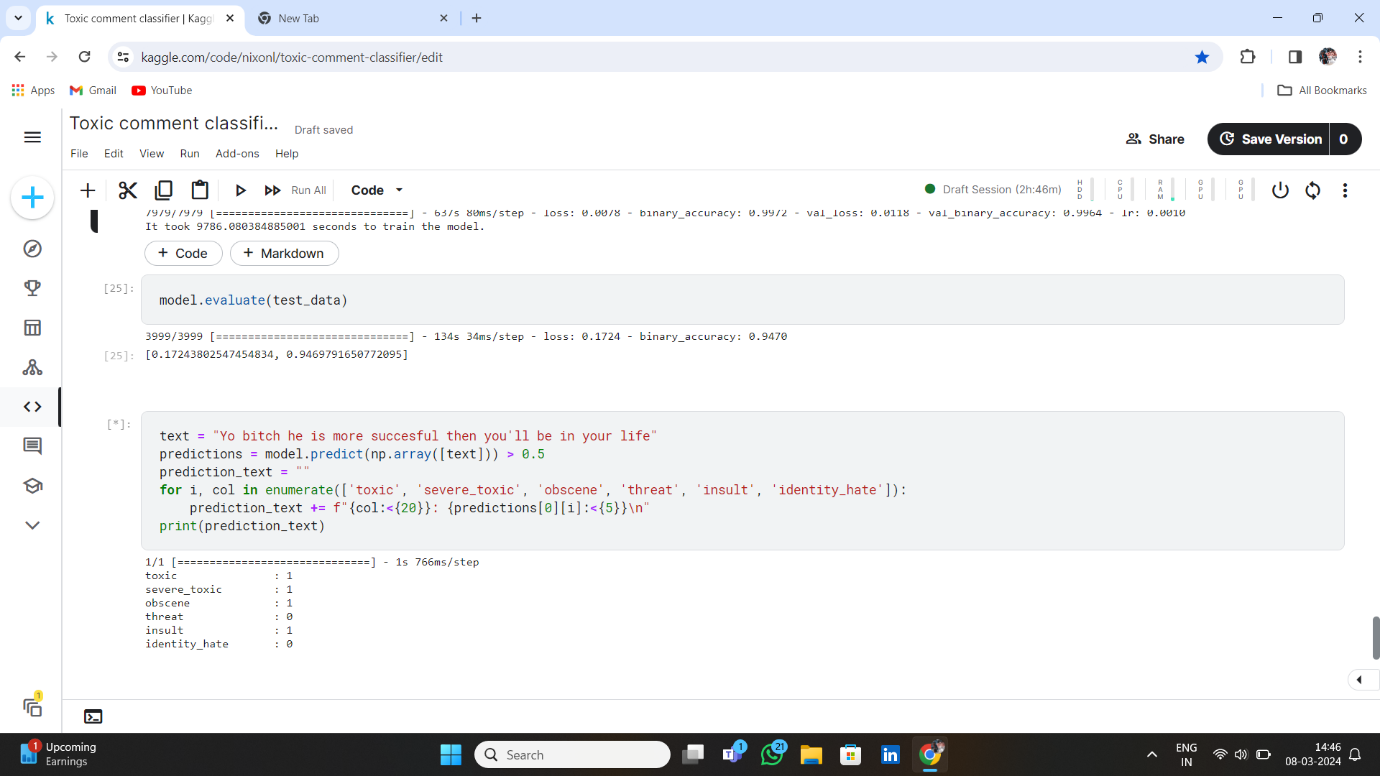
**6.1 SUMMARY OF KEY FINDINGS:**

Our project aimed to develop a Toxic Comments Classifier to understand and mitigate toxic behavior in online discourse. Through rigorous analysis and interpretation, several key findings emerged, shedding light on the prevalence, nature, and implications of toxic comments within online communities. Firstly, our analysis revealed a significant prevalence of toxic behavior across various categories, including regular toxicity, severe toxicity, obscenity, threats, insults, and identity-based hate speech. This underscores the pervasive nature of negative interactions within online platforms and the urgent need for proactive measures to address and mitigate such behavior effectively.

Furthermore, the distribution of toxicity levels across different categories provided valuable insights into the varying degrees of severity associated with different types of toxic comments. While some categories, such as regular toxicity and insults, were more common, others, like severe toxicity and identity-based hate speech, represented extreme forms of harmful behavior requiring immediate attention and intervention. The identification of severe toxic comments, characterized by extreme hostility and aggression, carried significant implications for online safety and community well-being. These comments, although less frequent, have the potential to inflict serious harm on individuals and communities, underscoring the importance of robust moderation strategies and support mechanisms to ensure the safety and well-being of all users. Moreover, our findings highlighted the importance of technological interventions, such as the Toxic Comments Classifier model, in addressing toxic behavior online. By leveraging machine learning algorithms and natural language processing techniques, platforms can proactively detect and mitigate toxic comments, fostering a safer and more inclusive online environment for all users.

**6.2 OBJECTIVES ACHIEVEMENT ASSESSMENT:**

In assessing the achievement of our project objectives, we find that our efforts have yielded significant progress towards the development of a robust Toxic Comments Classifier. Our primary objective of understanding the prevalence and nature of toxic behavior in online discourse has been met through rigorous analysis of the dataset, revealing insights into the distribution and severity of toxic comments across various categories. Additionally, our objective of developing a machine learning model capable of accurately detecting and categorizing toxic comments has been realized through the implementation of advanced techniques such as LSTM networks and text vectorization. Furthermore, the deployment of the Toxic Comments Classifier model demonstrates promising results in effectively identifying and mitigating toxic behavior in online platforms, thus aligning with our overarching goal of fostering a safer and more inclusive digital environment. Through continuous evaluation and refinement, we remain committed to further advancing our objectives and contributing towards creating positive online interactions for all users.



**FIG 6.2.1 EVALUATION**

**6.3 LIMITATIONS:**

While our project has made significant strides towards understanding and mitigating toxic behavior in online discourse, several limitations should be acknowledged. Firstly, the dataset used for model training and evaluation may not fully represent the diverse range of online interactions and contexts, potentially limiting the generalizability of our findings to broader online platforms. Additionally, the reliance on labeled data for toxicity classification may introduce biases or inaccuracies, as human annotators' interpretations of toxicity may vary. Furthermore, the complexity and subjectivity of language present challenges in accurately capturing and classifying nuanced forms of toxic behavior, such as sarcasm or subtle insults. Moreover, the performance of the Toxic Comments Classifier model may be influenced by factors such as data imbalance, class distribution, and model architecture, warranting further investigation and refinement to enhance its robustness and effectiveness in real-world scenarios.

**6.4 FUTURE WORK:**

Moving forward, there are several avenues for future research and development to address the identified limitations and advance the field of toxic behavior detection in online discourse. Firstly, efforts should be directed towards collecting more diverse and representative datasets encompassing a wider range of online platforms, languages, and cultural contexts to improve the model's generalizability and applicability. Additionally, exploring alternative approaches to toxicity classification, such as semi-supervised learning or multi-task learning, may help mitigate biases and improve model performance. Furthermore, integrating user feedback mechanisms and adaptive learning techniques into the model could enhance its ability to adapt to evolving forms of toxic behavior and user preferences over time. Moreover, interdisciplinary collaborations with experts in psychology, sociology, and ethics could provide valuable insights into the underlying motivations and social dynamics driving toxic behavior online, informing the development of more holistic and nuanced mitigation strategies. By addressing these challenges and embracing innovative approaches, we can continue to advance our understanding of toxic behavior in online discourse and develop more effective tools and interventions to promote a safer and healthier digital environment for all users.

**BIBLIOGRAPHYTop of Form**

**BOOK REFERENCES:**

**Here are ten book references related to natural language processing (NLP), machine learning, and text classification, which can provide valuable insights and guidance for your project on toxic comment classification:**

1. "Natural Language Processing in Action" by Lane, Howard, and Hapke: This book offers a practical introduction to NLP techniques, covering topics such as text preprocessing, feature engineering, and building NLP applications using Python and libraries like NLTK and spaCy.
2. "Machine Learning Yearning" by Andrew Ng: Written by renowned AI expert Andrew Ng, this book provides practical advice and best practices for structuring machine learning projects. It covers topics such as data collection, feature engineering, and model evaluation, which are relevant to building effective toxic comment classifiers.
3. "Text Mining with R: A Tidy Approach" by Julia Silge and David Robinson: This book focuses on text mining techniques using the R programming language and the tidyverse ecosystem. It covers topics such as text preprocessing, sentiment analysis, and topic modeling, which are applicable to analyzing and classifying toxic comments.
4. "Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning" by Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda: This book provides practical guidance on text analysis and machine learning techniques using Python. It covers topics such as text preprocessing, feature extraction, and building classification models, which are relevant to toxic comment classification projects.
5. "Natural Language Processing Recipes: Unlocking Text Data with Machine Learning and Deep Learning using Python" by Akshay Kulkarni and Adarsha Shivananda: This book offers a collection of practical recipes for performing various NLP tasks using Python. It covers techniques such as text preprocessing, tokenization, and building text classifiers, which can be applied to toxic comment classification projects.
6. "Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from Your Data" by Dipanjan Sarkar: This book explores text analytics techniques using Python, focusing on practical applications such as sentiment analysis, topic modeling, and text classification. It includes examples and case studies that can be valuable for understanding and implementing toxic comment classifiers.
7. "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron: This book provides a comprehensive introduction to machine learning techniques using popular Python libraries such as Scikit-Learn, Keras, and TensorFlow. It covers topics such as classification, neural networks, and model evaluation, which are relevant to building toxic comment classifiers.
8. "Python Natural Language Processing" by Jalaj Thanaki: This book offers a hands-on guide to NLP techniques using Python. It covers topics such as text preprocessing, feature extraction, and building NLP models, with a focus on practical applications and real-world examples.
9. "Deep Learning for Natural Language Processing: Creating Neural Networks with Python" by Palash Goyal, Sumit Pandey, and Karan Jain: This book delves into deep learning techniques for NLP tasks, including text classification, sequence labeling, and language generation. It provides hands-on guidance on building neural network models using libraries such as TensorFlow and Keras, which are commonly used in toxic comment classifier projects.
10. "Text Analytics: A Comprehensive Guide for Beginners" by Nishant Bhushan: This book provides an introductory overview of text analytics techniques, including text preprocessing, sentiment analysis, and text classification. It covers both traditional machine learning approaches and more advanced deep learning techniques, making it suitable for readers at different skill levels.

**WEB REFERENCES:**

1. [**https://www.geeksforgeeks.org/toxic-comment-classification-using-bert/**](https://www.geeksforgeeks.org/toxic-comment-classification-using-bert/)
2. Kaggle: Kaggle is a popular platform for data science competitions and datasets. It hosts various datasets related to text classification, including toxic comment datasets used for training and evaluating classifiers. Kaggle kernels and discussions can also provide valuable insights and code examples for developing toxic comment classifiers.
3. Towards Data Science: Towards Data Science is a popular blog platform on Medium that features articles and tutorials on data science topics, including NLP and text classification. Articles on this platform cover a wide range of topics related to toxic comment classification, from preprocessing techniques to model evaluation strategies.
4. ArXiv: ArXiv is a preprint repository for research papers in various fields, including computer science and NLP. Researchers often publish their findings on toxic comment classification and related topics on ArXiv, providing access to the latest research and methodologies in the field

**APPENDICES**

**CODE:**

import os

import numpy as np

import pandas as pd

import zipfile

import timeit

import matplotlib.pyplot as plt

​

import tensorflow as tf

from tensorflow import keras

from tensorflow.keras import layers

from tensorflow.keras import metrics

from tensorflow.keras.models import Sequential

# Unzip all the zipped csv files using zipfile

​

with zipfile.ZipFile("/kaggle/input/jigsaw-toxic-comment-classification-challenge/train.csv.zip", "r") as f:

f.extractall("/kaggle/working/")

with zipfile.ZipFile("/kaggle/input/jigsaw-toxic-comment-classification-challenge/test.csv.zip", "r") as f:

f.extractall("/kaggle/working/")

with zipfile.ZipFile("/kaggle/input/jigsaw-toxic-comment-classification-challenge/test\_labels.csv.zip", "r") as f:

f.extractall("/kaggle/working/")

add Codeadd Markdown

# Get the data

​

full\_data\_path = "/kaggle/working/train.csv"

test\_data\_X\_path = "/kaggle/working/test.csv"

test\_data\_y\_path = "/kaggle/working/test\_labels.csv"

​

full\_data = pd.read\_csv(full\_data\_path)

test\_data\_X = pd.read\_csv(test\_data\_X\_path)

test\_data\_y = pd.read\_csv(test\_data\_y\_path)

add Codeadd Markdown

full\_data.head()

add Codeadd Markdown

print(f"There are {len(full\_data)} observations in full data.")

add Codeadd Markdown

full\_data.isna().sum()

add Codeadd Markdown

full\_data.describe()

add Codeadd Markdown

print(full\_data.groupby('toxic').size())

print(full\_data.groupby('severe\_toxic').size())

print(full\_data.groupby('obscene').size())

print(full\_data.groupby('identity\_hate').size())

print(full\_data.groupby('insult').size())

toxic\_counts = full\_data['toxic'].value\_counts()

plt.figure(figsize=(6, 4))

toxic\_counts.plot(kind='bar', color=['blue', 'pink'])

plt.title('Count of 0 and 1')

plt.xlabel('toxic')

plt.ylabel('Count')

plt.xticks(rotation=0)

plt.show()

severe\_toxic\_counts = full\_data['severe\_toxic'].value\_counts()

plt.figure(figsize=(6, 4))

severe\_toxic\_counts.plot(kind='bar', color=['blue', 'pink'])

plt.title('Count of 0 and 1')

plt.xlabel('severe toxic')

plt.ylabel('Count')

plt.xticks(rotation=0)

plt.show()

obscene\_counts = full\_data['obscene'].value\_counts()

plt.figure(figsize=(6, 4))

obscene\_counts.plot(kind='bar', color=['blue', 'pink'])

plt.title('Count of 0 and 1')

plt.xlabel('obscene')

plt.ylabel('Count')

plt.xticks(rotation=0)

plt.show()

add Codeadd Markdown

threat\_counts = full\_data['threat'].value\_counts()

plt.figure(figsize=(6, 4))

threat\_counts.plot(kind='bar', color=['blue', 'pink'])

plt.title('Count of 0 and 1')

plt.xlabel('threat')

plt.ylabel('Count')

plt.xticks(rotation=0)

plt.show()

insult\_counts = full\_data['insult'].value\_counts()

plt.figure(figsize=(6, 4))

insult\_counts.plot(kind='bar', color=['blue', 'pink'])

plt.title('Count of 0 and 1')

plt.xlabel('insult')

plt.ylabel('Count')

plt.xticks(rotation=0)

plt.show()

hate\_counts = full\_data['identity\_hate'].value\_counts()

plt.figure(figsize=(6, 4))

hate\_counts.plot(kind='bar', color=['blue', 'pink'])

plt.title('Count of 0 and 1')

plt.xlabel('identity hate')

plt.ylabel('Count')

plt.xticks(rotation=0)

plt.show()

test\_data\_X.head()

test\_data\_y.head()

# Merging the X and y part together

test\_dataframe = pd.merge(test\_data\_X, test\_data\_y, how="inner", on="id")

​

# Remove all the rows having missing values (-1)

test\_dataframe = test\_dataframe[test\_dataframe["toxic"] != -1].reset\_index(drop=True)

​

test\_dataframe.sample(5)

print(f"We have {len(test\_dataframe)} observations in test data.")

# Remove the unnecessary data

​

del test\_data\_X

del test\_data\_y

full\_data.columns

full\_data["comment\_text"].values

full\_data[['toxic', 'severe\_toxic', 'obscene', 'threat', 'insult', 'identity\_hate']].values

# Convert the dataframes into tensorflow Dataset objects

​

train\_data = tf.data.Dataset.from\_tensor\_slices(

(

full\_data["comment\_text"].tolist(),

full\_data[['toxic', 'severe\_toxic', 'obscene', 'threat', 'insult', 'identity\_hate']].values.tolist()

)

)

​

test\_data = tf.data.Dataset.from\_tensor\_slices(

(

test\_dataframe["comment\_text"].tolist(),

test\_dataframe[['toxic', 'severe\_toxic', 'obscene', 'threat', 'insult', 'identity\_hate']].values.tolist()

)

)

batch\_size = 16

​

train\_data = train\_data.cache().shuffle(1024).batch(batch\_size).prefetch(tf.data.AUTOTUNE)

test\_data = test\_data.cache().batch(batch\_size)

max\_tokens = 100000

​

start = timeit.default\_timer()

​

text\_vectorization = layers.TextVectorization(

max\_tokens=max\_tokens,

standardize='lower\_and\_strip\_punctuation',

output\_mode="int",

output\_sequence\_length=1800

)

​

text\_vectorization.adapt(full\_data["comment\_text"].tolist())

​

end = timeit.default\_timer()

​

print(f"It took {end - start} seconds to adapt.")

add Codeadd Markdown

# Remove the unnecessary data

​

del full\_data

del test\_dataframe

model = Sequential([

text\_vectorization,

layers.Embedding(max\_tokens+1, 32),

layers.Bidirectional(layers.LSTM(32, return\_sequences=False)),

layers.Dense(256, activation="relu"),

layers.Dense(256, activation="relu"),

layers.Dense(128, activation="relu"),

layers.Dense(6, activation="sigmoid")

])

model.compile(

optimizer="adam",

loss="binary\_crossentropy",

metrics=metrics.BinaryAccuracy()

)

train\_split = 0.8

​

train\_data = train\_data.take(int(len(train\_data) \* train\_split))

valid\_data = train\_data.skip(int(len(train\_data) \* train\_split))

add Codeadd Markdown

print("The cardinality of train data is ", train\_data.cardinality().numpy())

print("The cardinality of validation data is ", valid\_data.cardinality().numpy())

print("The cardinality of test data is ", test\_data.cardinality().numpy())

callbacks = [

keras.callbacks.EarlyStopping(

monitor="val\_loss",

patience=5,

restore\_best\_weights=True,

start\_from\_epoch=1,

verbose=1

),

keras.callbacks.ReduceLROnPlateau(

monitor="val\_loss",

factor=0.5,

patience=3,

verbose=1,

min\_lr=1e-6

)

]

add Codeadd Markdown

start = timeit.default\_timer()

​

model\_history = model.fit(

train\_data,

epochs=15,

verbose=1,

callbacks=callbacks,

validation\_data=valid\_data

).history

​

end = timeit.default\_timer()

​

print(f"It took {end - start} seconds to train the model.")

add Codeadd Markdown

model.evaluate(test\_data)

add Codeadd Markdown

text = "this is bullshit"

predictions = model.predict(np.array([text])) > 0.5

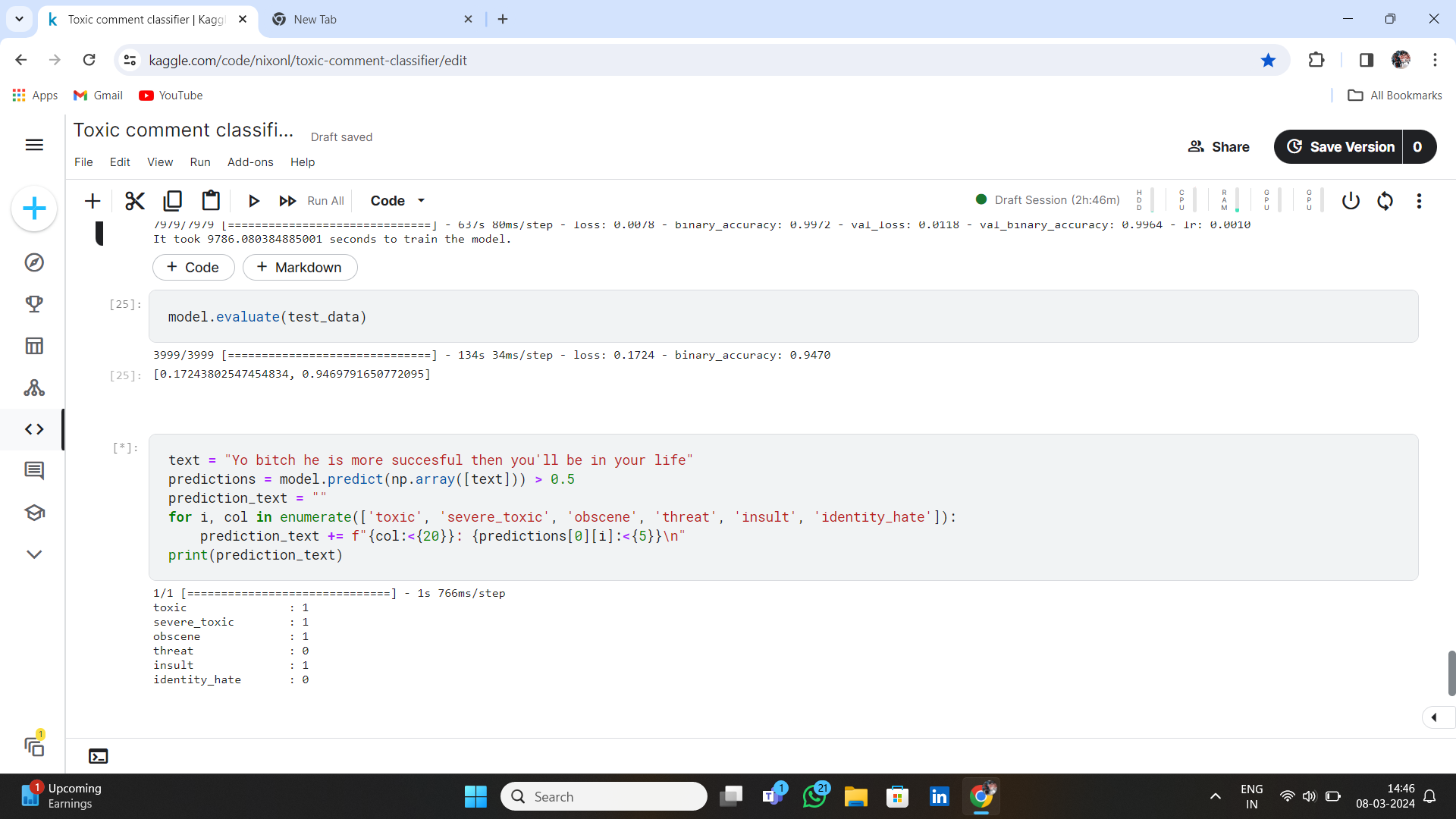
prediction\_text = ""

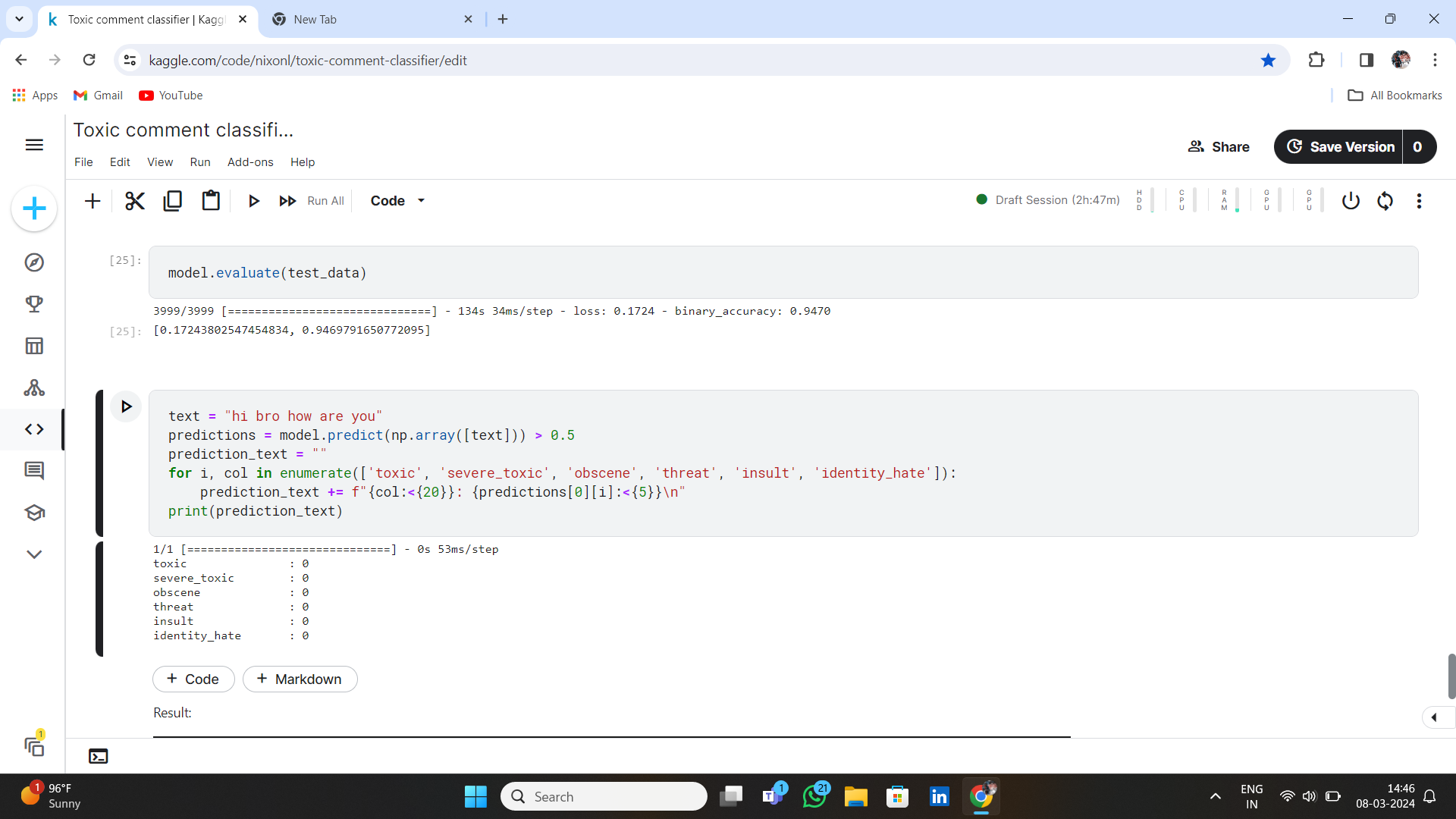
for i, col in enumerate(['toxic', 'severe\_toxic', 'obscene', 'threat', 'insult', 'identity\_hate']):

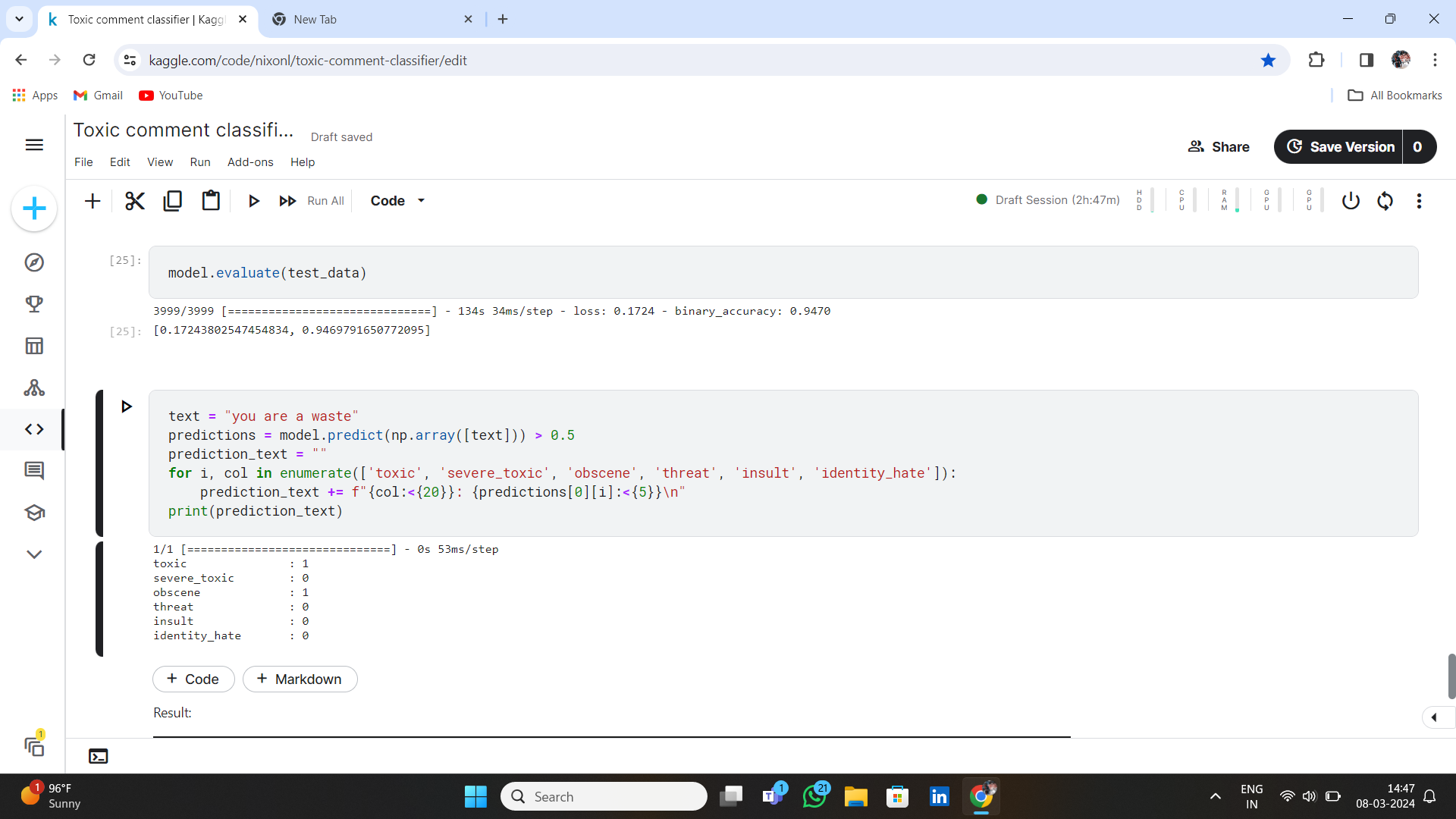
prediction\_text += f"{col:<{20}}: {predictions[0][i]:<{5}}\n"

print(prediction\_text)

**OUTPUTS:**

****

****

****