**Final Report**

**Reddit Sarcasm Prediction**

**Team Members:**

1. **Mayankkumar Patel**
2. **Jasleen Kaur**
3. **Komal Astawala**
4. **Shree Prada**
5. **Bhavneet Kaur**
6. **Simran**

**1. Git Implementation for Reddit Sarcasm Prediction:**

In our Reddit Sarcasm Prediction project, Git is the backbone of our development process, ensuring smooth collaboration and effective version control. Here's how we leverage Git**:**

**Organized Repository:** We maintain a well-structured repository with directories like Flask\_Sarcasm, PKL file each dedicated to specific project components.

**Collaborative Workflow:** Git's branching feature allows team members to work independently on features or bug fixes. We create branches for each task and merge them back into the main branch after testing.

**Version Control:** Git meticulously tracks every change made to our codebase. With descriptive commit messages, we document the purpose and context of each modification, facilitating easy rollback if needed.

**Collaboration Tools:** We utilize Git's pull request feature for proposing and reviewing code changes. This ensures that all contributions undergo thorough review before integration into the main codebase.

**Branching Strategy:** Adopting a structured branching strategy helps us maintain a clean and organized codebase. Feature branches are created for implementing new functionalities, while bug fix branches address issues promptly.

**2. Model Development:**

The model development phase of the Reddit Sarcasm Prediction project revolves around crafting and training a machine learning model capable of accurately discerning sarcasm within Reddit comments. This segment presents an overview of the dataset employed, preprocessing methodologies, model architecture, training regimen, and evaluation metrics.

**2.1 Dataset:** The project harnesses a meticulously curated dataset comprising labeled Reddit comments, where each comment is annotated as either sarcastic or non-sarcastic.

**2.2 Preprocessing:** Text preprocessing constitutes a pivotal stride in readying the data for model training. Common techniques entail:

* **Text cleaning:** Punctuation, special characters, Emoji handling, Replacing Abbreviations etc.
* **Tokenization:** Breaking text into individual words or phrases.
* **Stopword removal:** Eliminating common words that carry little meaning.
* **Lemmatization:** Reducing words to their base or dictionary form.

**Feature engineering includes:**

* **Word embeddings(GloVe):** Capturing semantic relationships between words in a continuous vector space.
* **TD-IDF Vectorization:** Capturing important words for each data points in the Corpus.
* **Combined matrix:** TF-IDF + Glove embeddings concat

**2.3 Model Architecture:** The sarcasm prediction model typically leverages a neural network architecture, such as Long Short-Term Memory (LSTM). Salient aspects include:

LSTM networks adeptly handle sequential data processing tasks like text classification, capturing temporal dependencies within input sequences.

**2.5 Evaluation Metrics:** The efficiency of the sarcasm prediction model is gauged through diverse metrics, including accuracy, precision, recall, and F1 score. Key metrics include:

* Best LSTM Accuracy on Training Set: 0.998875
* Best LSTM Accuracy on Test Set: 0.629
* Precision: 0.6458865610628512
* Recall: 0.6150851581508516
* F1-score: 0.6301096709870389
* Confusion Matrix: [[1252  693] [ 791 1264]]

**3. Model Tuning:**

Custom LSTM Model:

* In order to apply GridSearch cross validation for different number of epochs we have made our own custom LSTM classifier estimator and ran our model with 20 and 50 epochs in order to find the best parameters.
* We have used 5 fold cross validation

Custom LSTM based SVM:

* In order to apply GridSearch cross validation for parameters c, Gamma, and Kernel to find the best parameters.
* We have used 5 fold cross validation

**4. Model Deployment - Flask:**

Deploying the trained sarcasm prediction model is a pivotal milestone in our Reddit Sarcasm Prediction project, facilitating real-time inference through a user-friendly web interface. Leveraging Flask, a lightweight Python web framework, we seamlessly integrate the model into an API endpoint, enabling users to interact with it via HTTP requests. This section elucidates the model deployment process using Flask and elucidates the steps involved in crafting an intuitive interface for sarcasm prediction.

**4.1 Setting up Flask:** Flask furnishes an agile framework for building web applications and APIs in Python, initiating the model deployment journey. We kickstart the process by creating a Python script, such as app.py, wherein we define routes to handle HTTP requests. These routes, adorned with decorators like @app.route('/'), delineate URL paths and execute corresponding functions upon access.

**4.2 Integrating the Model:** The crux of model deployment lies in seamlessly integrating the trained sarcasm prediction model into our Flask application. This entails loading model and defining functions to execute inference on user inputs. Subsequently, the Flask route dedicated to prediction requests invokes the model's inference function, passing user input as input data.

**4.3 Creating the User Interface:** To enhance user experience, we craft a bespoke web interface for sarcasm prediction, blending HTML, CSS, and JavaScript. This interface typically comprises an input form where users input Reddit comments and a button to initiate the prediction process. JavaScript orchestrates user input capture, AJAX request dispatch to the Flask backend, and dynamic display of prediction results on the web page.

**5. Demo: Custom UI:**

The demonstration of the Reddit Sarcasm Prediction project showcases a custom user interface (UI) designed to provide users with an intuitive and interactive platform for predicting sarcasm in Reddit comments. This section details the features and functionality of the custom UI, along with a walkthrough of its usage.

**5.1 Interface Overview:** The custom UI features a clean and user-friendly design, with intuitive controls and informative feedback to guide users through the prediction process.

**The main components of the UI include:**

**Input Text Box:** Allows users to enter Reddit comments for sarcasm prediction.

**Prediction Button:** Triggers the prediction process when clicked.

**Prediction Result:** Displays the model's prediction (sarcastic or non-sarcastic) based on the input comment.

**5.2 Input and Prediction Process:** Users begin by entering a Reddit comment into the input text box, simulating a real-world scenario where users provide text for sarcasm prediction.

Upon entering the comment, users click the prediction button to initiate the prediction process.

The UI sends the entered comment to the Flask backend via an HTTP request, where the sarcasm prediction model performs inference on the input text.

**5.3 Model Inference and Result Display:** In the backend, the Flask application receives the user input and passes it to the trained sarcasm prediction model for inference.

The model processes the input text and generates a prediction (sarcastic or non-sarcastic) based on learned patterns and features.

The prediction result is returned to the UI and displayed to the user, providing instant feedback on the sarcasm status of the input comment.

**6. Final Results:**

Our model demonstrates robust performance in distinguishing between sarcastic and non-sarcastic comments.

**7. Project Pitch**

* Our project aims to address the challenge of sarcasm detection in online conversations, a crucial task in sentiment analysis and social media monitoring.
* By accurately predicting sarcasm, our model can aid in sentiment analysis, content moderation, and brand reputation management.

Potential applications include:

* Social media analytics tools
* Customer feedback analysis platforms
* Online community moderation systems

**8. Learnings:**

Key learnings from the project include:

* Deepening understanding of NLP techniques and methodologies.
* Gaining practical experience in model development, tuning, and deployment.
* Overcoming challenges in data preprocessing, training, and evaluation.
* Insights gained contribute to the continuous improvement and innovation in NLP research and application.

**9.Target audience:**

Our project caters to a diverse audience including:

* Data scientists and NLP researchers interested in sarcasm detection and sentiment analysis.
* Social media platform developers seeking to enhance content moderation algorithms.
* Marketing professionals aiming to gauge audience sentiment and engagement on digital platforms.

**MECE Table:**

| **Category** | **Subcategory** | **Owner** |
| --- | --- | --- |
| **Data Collection** | Kaggle Reddit Dataset | Data Team |
| **Data Preprocessing** | Text Cleaning | Shree Prada |
|  | Tokenization | Shree Prada |
|  | Abbreviation Expansion | Komal |
|  | Slang Translation | Mayank |
|  | Stopwords Removal | Simran |
|  | Lemmatization | Jasleen |
| **Feature Engineering** | GloVe Embeddings | Komal |
|  | TF-IDF Vectors | Shree Prada |
|  | Combined Matrix | Mayank |
| **Modeling** | Custom LSTM | Mayank |
|  | Custom LSTM based SVM | Komal |
| **Evaluation** | Accuracy | Bhavneet |
|  | Precision | Bhavneet |
|  | Recall | Simran |
|  | F1 Score | Simran |
| **Interpretation** | SHAP | Mayank |
| **Deployment** | Flask App | Jasleen |
|  | UI Design | Shree Prada |