NUST

NATIONAL UNIVERSITY
OF SCIENCES & TECHNOLOGY

PROJECT REPORT

MACHINE LEARNING

NAME	CMS ID
Laiba Atiq	372187
Abdullah Tahir	393056

Instructor: Dr Daud Abdullah

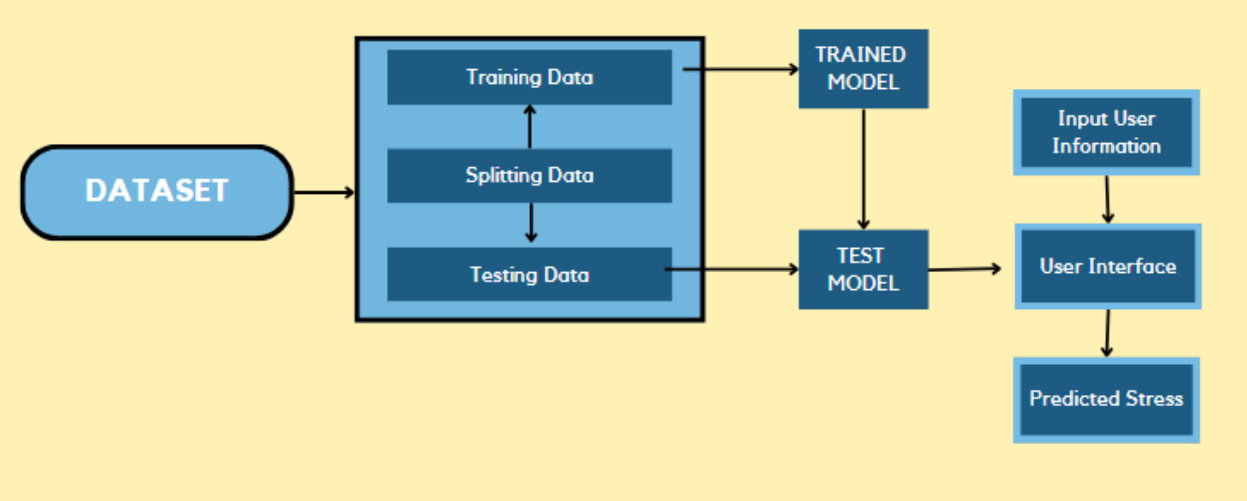
CONTENTS

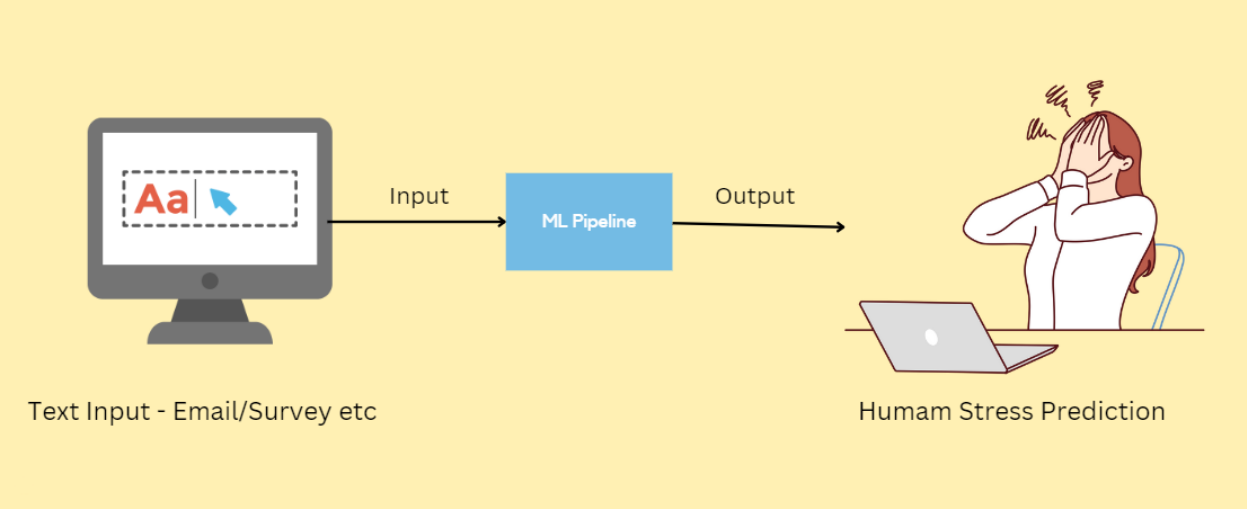
STRESS PREDICTION USING TEXTUAL DATA

INTRODUCTION:

Stress is a pervasive issue in modern life, affecting our well-being and productivity. To address this challenge, we've developed a unique stress prediction tool. It leverages the power of machine learning to analyze how people express their feelings in text, detecting signs of stress with remarkable accuracy. Imagine integrating this tool into a workplace setting – it could assess employees' well-being before they even log in, offering personalized advice to promote a healthier work environment. In the future, this system could proactively signal when someone consistently shows signs of stress, allowing us to offer timely support and prevent burnout. This project aims to harness the power of AI to not just understand stress, but to create a more proactive and compassionate way to support those impacted by it.

WORKFLOW:





To develop the stress predictor, we started by sourcing a dataset from Kaggle containing text where people express their feelings. This text data was cleaned and preprocessed using techniques like tokenization, stop word removal, and stemming/lemmatization. Next, we transformed the text into numerical features suitable for machine learning algorithms. We then trained and evaluated various classification algorithms (Naive Bayes, SVM, Random Forest, etc.) to identify the best model for stress prediction. This model has the potential for integration into workplace environments, enabling proactive well-being assessments and offering timely stress alerts.

DATASET USED:

The dataset used for this stress prediction model consists of text excerpts sourced from Reddit posts across various subreddits e.g ptsd, relationships, etc. It aims to capture expressions of feelings and potential indicators of stress. Here's a description of the key features:

- **subreddit:** (Nominal)

  Indicates the subreddit where the post originated.

- **post_id:** (Nominal)

  A unique identifier for the Reddit post.

- **sentence_range:** (Numerical)

  Represents the length of the post extracted.

- **text:** (Text)

  The actual textual content of the post excerpt.

- **id:** (Numerical)

  Unique identifier for each excerpt.

- **label:** (Binary)

  The target variable indicating stress presence - '1' for stressed, '0' for not stressed.

- **confidence:** (Numerical)

  A score from 0-1

- **social_timestamp:** (Numerical)

  Unix timestamp when the post was made.

- **social_karma:** (Numerical)

  Reddit user reputation score.

- **syntax_ari:** (Numerical)

  Automated Readability Index - a measure of the text's complexity.

- **Lexical analysis columns:** (Numerical)

  Likely sentiment-related scores based on a lexicon for the post text.

The data was originally like this:

```
          subreddit post_id sentence_range  \
0              ptsd  8601tu       (15, 20)
1       assistance  8lbrx9         (0, 5)
2              ptsd  9ch1zh       (15, 20)
3     relationships  7rorpp        [5, 10]
4  survivorsofabuse  9p2gbc         [0, 5]

                                              text     id  label  \
0  He said he had not felt that way before, sugge...  33181      1
1  Hey there r/assistance, Not sure if this is th...   2606      0
2  My mom then hit me with the newspaper and it s...  38816      1
3  until i met my new boyfriend, he is amazing, h...    239      1
4  October is Domestic Violence Awareness Month a...   1421      1

   confidence  social_timestamp  social_karma  syntax_ari  ...  \
0         0.8        1521614353             5    1.806818  ...
1         1.0        1527009817             4    9.429737  ...
2         0.8        1535935605             2    7.769821  ...
3         0.6        1516429555             0    2.667798  ...
4         0.8        1539809005            24    7.554238  ...

   lex_dal_min_pleasantness  lex_dal_min_activation  lex_dal_min_imagery  \
0                     1.000                  1.1250                  1.0
1                     1.125                  1.0000                  1.0
2                     1.000                  1.1429                  1.0
...
3                0.50                   5         4.104027   0.141671
4                1.00                   1         7.910952  -0.204167

[5 rows x 116 columns]
```

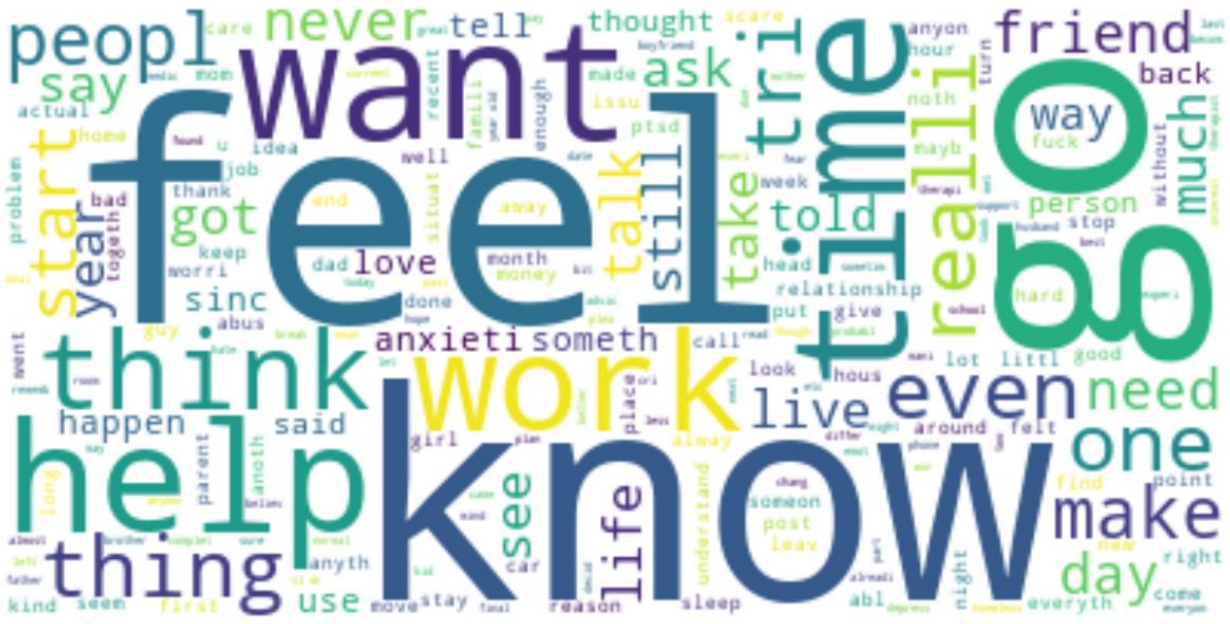
PREPROCESSING:

The data contained no null values:

```
Click here to ask Blackbox to help you code faster
print(data.isnull().sum())
✓ 0.0s

subreddit                    0
post_id                      0
sentence_range               0
text                         0
id                           0
                            ..
lex_dal_avg_pleasantness     0
social_upvote_ratio          0
social_num_comments          0
syntax_fk_grade              0
sentiment                    0
Length: 116, dtype: int64
```

Wordcloud showing the most common words used in the data are:



On analysis, it is seen that the classification approach we are using does not need any of the original columns like confidence, lexical analytics, subreddit name, social karma of the user etc. So, we removed all the columns of the datasets except the labels and the text. The labels were replaced with Stress or No Stress for better understanding.

```
Click here to ask Blackbox to help you code faster
data["label"] = data["label"].map({0: "No Stress", 1: "Stress"})
data = data[["text", "label"]]
print(data.head())
```

✓ 0.0s

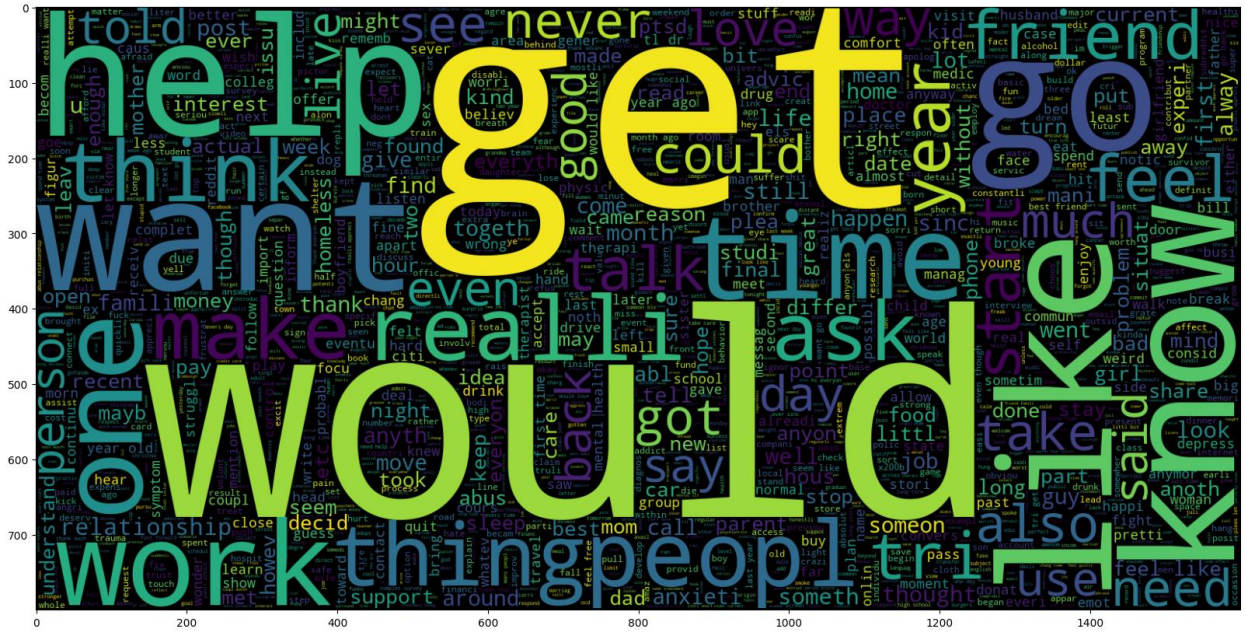
	text	label
0	said felt way sugget go rest trigger ahead you...	Stress
1	hey r assist sure right place post goe current...	No Stress
2	mom hit newspap shock would know like play hit...	Stress
3	met new boyfriend amaz kind sweet good student...	Stress
4	octob domest violenc awar month domest violenc...	Stress

DATA CLEANING NLP:

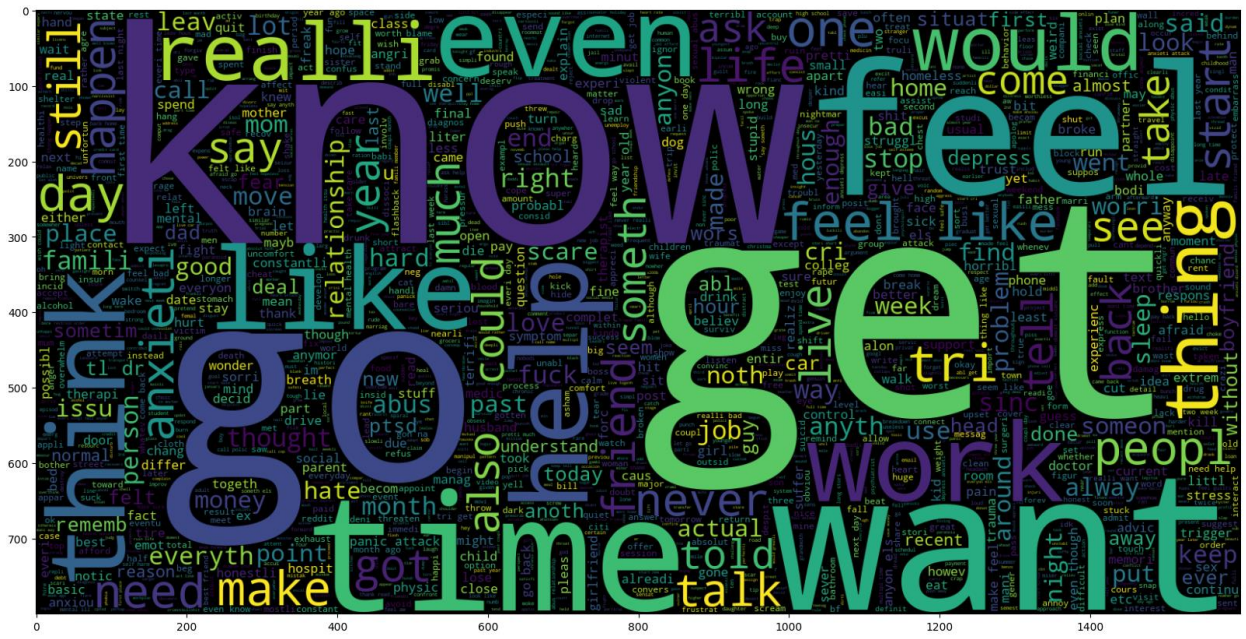
We employed several text cleaning and preprocessing techniques to prepare the raw Reddit data for our stress prediction model. First, we converted all text to lowercase to ensure consistency and reduce noise. Then, we removed brackets, URLs, escape characters, and HTML tags, all of which could interfere with accurate analysis. Next, we filtered out non-alphanumeric characters to focus on words. To further refine the text, we used NLTK (Natural Language Toolkit) to tokenize the sentences into individual words, remove common stop words (like "the", "and"), and apply stemming using the PorterStemmer. Stemming reduces words to their base form, improving the model's ability to recognize related terms (e.g., "running" and "run" would be mapped to the same root "run"). This comprehensive cleaning process aimed to isolate the core meaning within the text, making it more suitable for our machine learning model.

EXPLORATORY DATA ANALYSIS (EDA)

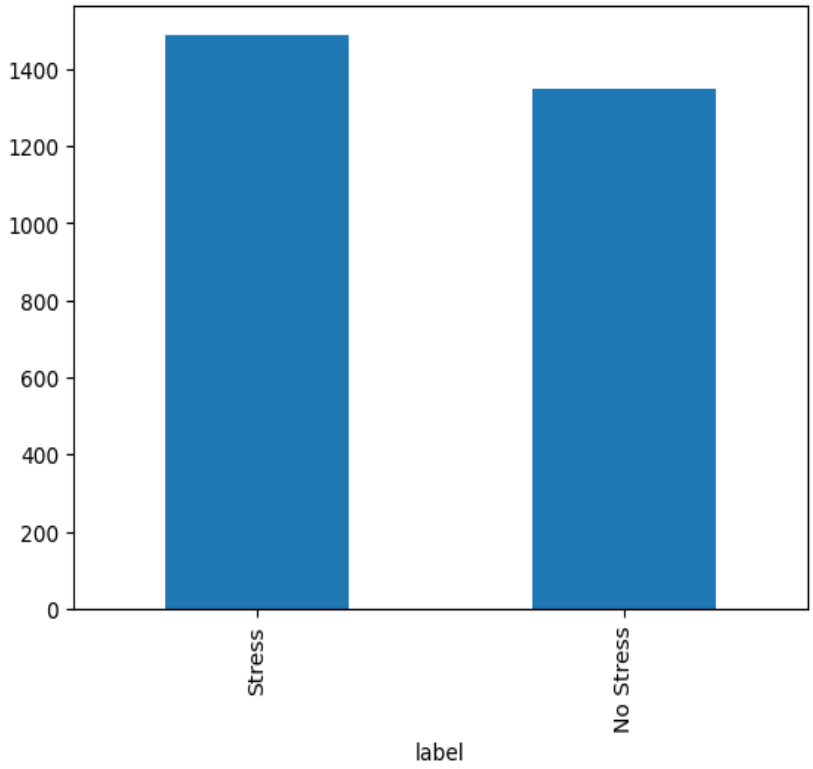
This word cloud emphasizes terms frequently occurring in sentences labeled as "no stress" within the dataset. Words like "help," "good," and "friend" appear prominently, reflecting a sense of peace and well-being. The visual reinforces the positive sentiment associated with the "no stress" category and aids in understanding the language patterns indicative of a stress-free state.



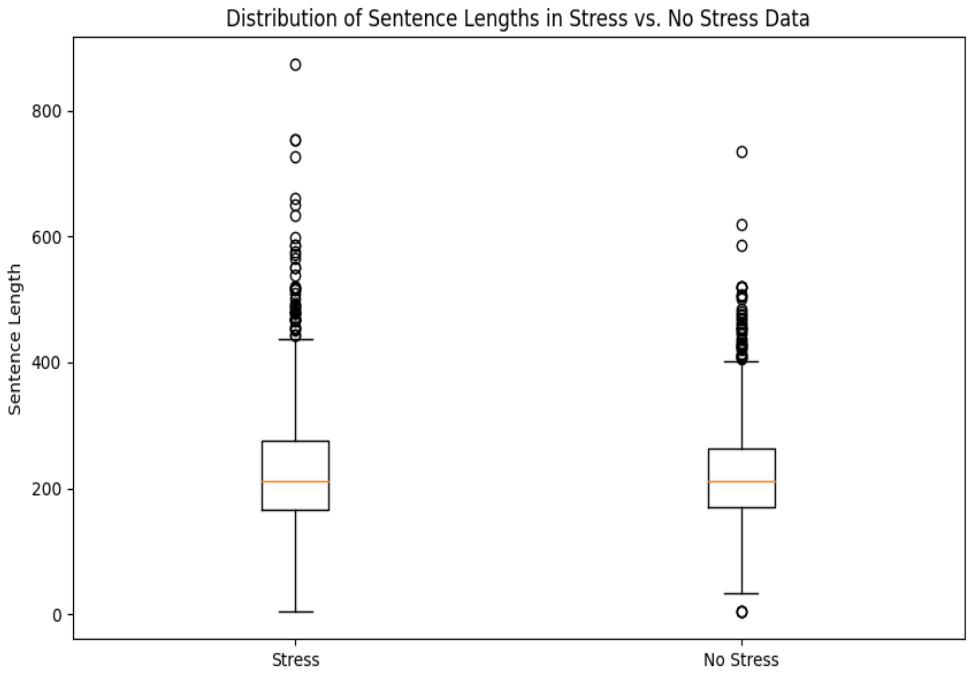
To complement the analysis of non-stressful language, a word cloud visualization was created for the "stress" category. This word cloud highlights words frequently appearing in sentences labeled as "stressful" within the dataset. By analyzing the frequency of these terms, we gain insights into the language patterns indicative of stress and how people express themselves when feeling stressed. This visualization can further aid in refining the stress prediction model by identifying the most significant keywords associated with stress.



The bar graph shows the comparative number of no stress and stress labels in the datasets. It shows that although the stress labels are more, there is not a huge difference.



The box and whisker plot shows the analysis of lengths of the sentences in Stress and no stress conditions. This basically shows how the length of the sentence cannot be a factor in determining as the plots are more or less the same for both.



FEATURE EXTRACTION WITH TF-IDF:

In our approach to stress prediction, we opted to employ TF-IDF vectorization over count vectorization to transform textual data into numerical features for machine learning. This technique goes beyond simple word counts by incorporating a valuable concept: term importance. TF-IDF considers both how frequently a word appears within a single text entry (term frequency) and its rarity across the entire dataset (inverse document frequency). This focus on relative importance is crucial for my analysis. Words that frequently appear in stressed or non-stressed examples will be downplayed, as they likely hold less discriminatory power for the model. Conversely, terms that are characteristic of stressed or non-stressed text but uncommon overall will be emphasized. This weighting helps the model learn the subtle nuances of language use that might signal stress and ultimately leads to a more robust and informative feature representation compared to a basic word count approach.

```
from sklearn.feature_extraction.text import TfidfVectorizer
tf = TfidfVectorizer(min_df=1)
tf_df = tf.fit_transform(x)
tf_df.toarray()
```

TRAIN/TEST SPLIT

We selected a 0.2 test size (meaning 80% of the data is for training, 20% for testing) as it proved optimal for our stress prediction system. This split strikes a crucial balance. A larger training set allows the model to better learn the complexities of stress expression within text, while the testing set remains large enough for a trustworthy evaluation of the model's ability to handle unseen data. Our experimentation with different splits confirmed that this specific ratio yielded the highest performance.

METHODOLOGY

We began by taking the textual data from Reddit from our Stress.csv file. Preprocessing involved cleaning text (removing non-alphanumeric characters, applying stopword filtering, and stemming), followed by visualizing frequently used words. To represent the text for machine learning, we used TF-IDF vectorization. This method highlights words that are discriminative for the "stress" and "no stress" categories. We then split the data into training and testing sets. Various classifiers (Support Vector Machines, Logistic Regression, Naive Bayes, Random Forest) were explored and hyperparameter tuning was performed using grid search to optimize performance. The best models were combined in a Voting Classifier. Finally, the system's performance was evaluated on unseen data using metrics like accuracy and a confusion matrix.

MODELS USED:

SUPPORT VECTOR MACHINE:

To enhance our stress detection model, we employed hyperparameter tuning on a Support Vector Machine (SVM) classifier. Using grid search, we methodically tested various combinations of kernels (e.g., 'linear', 'poly'), gamma values, and regularization parameters (`C`). This process helped us find the optimal configuration of hyperparameters, which ultimately yielded an accuracy of 73% during cross-validation.

```
[[190  73]
 [ 82 223]]
              precision    recall  f1-score   support

   No Stress       0.70      0.72      0.71       263
      Stress       0.75      0.73      0.74       305

    accuracy                           0.73       568
   macro avg       0.73      0.73      0.73       568
weighted avg       0.73      0.73      0.73       568
```

LOGISTIC REGRESSION

We employed hyperparameter tuning on a Logistic Regression classifier to enhance our stress detection model. Using grid search, we methodically explored various combinations of regularization strengths (C), optimization algorithms ('solver'), and maximum iteration limits ('max_iter'). This process enabled us to identify the optimal configuration, which ultimately achieved an accuracy of 74% during cross-validation.

```
[[186  77]
 [ 72 233]]
              precision    recall  f1-score   support

   No Stress       0.72      0.71      0.71       263
      Stress       0.75      0.76      0.76       305

    accuracy                           0.74       568
   macro avg       0.74      0.74      0.74       568
weighted avg       0.74      0.74      0.74       568
```

BERNOULLI NAIVE BAYES

To refine our stress detection model, we focused on tuning the alpha parameter within a Bernoulli Naive Bayes classifier. This alpha parameter controls the degree of smoothing applied during the probability estimation. We tested a range of alpha values and evaluated each resulting model on our testing set. Through this careful experimentation, we discovered that an alpha value of 1 yielded the best performance, resulting in a test set accuracy of 76%.

```
[[170  93]
 [ 46 259]]
              precision    recall  f1-score   support

   No Stress       0.79      0.65      0.71       263
      Stress       0.74      0.85      0.79       305

    accuracy                           0.76       568
   macro avg       0.76      0.75      0.75       568
weighted avg       0.76      0.76      0.75       568
```

RANDOM FOREST REGRESSION:

To optimize our Random Forest classifier for stress detection, we employed hyperparameter tuning. Using grid search, we systematically evaluated combinations of parameters controlling the number of trees (n_estimators), tree depth (max_depth), and splitting criteria (min_samples_split, min_samples_leaf). This process led us to identify the ideal configuration, which achieved a cross-validation accuracy of 0.7180616740088106. Our best model had the following parameters: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200}.

```
[[167  96]
 [ 59 246]]
              precision    recall  f1-score   support

   No Stress       0.74      0.63      0.68       263
      Stress       0.72      0.81      0.76       305

    accuracy                           0.73       568
   macro avg       0.73      0.72      0.72       568
weighted avg       0.73      0.73      0.72       568
```

MULTIMODAL NAIVE BAYES:

We experimented with hyperparameter tuning to enhance our Multinomial Naive Bayes stress detection model. Specifically, we focused on adjusting the alpha parameter, which controls smoothing during probability calculations. Using grid search, we tested several alpha values and assessed each configuration's performance through cross-validation. This exploration helped us pinpoint the optimal alpha value, leading to the best possible cross-validation score.

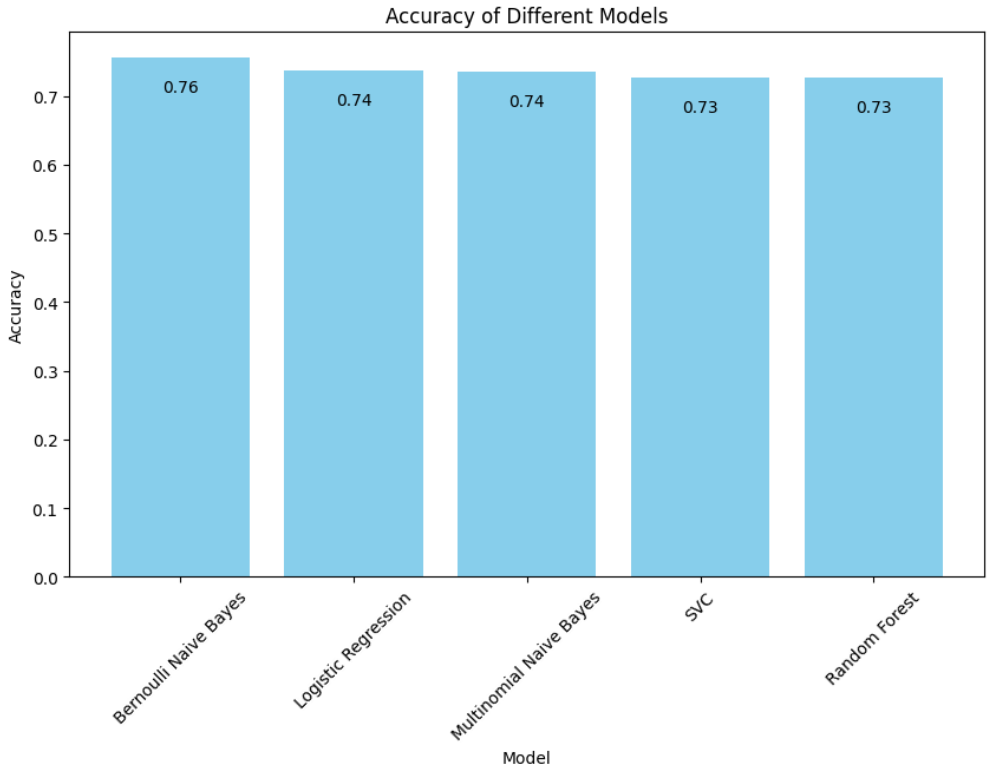
```
Best parameters: {'alpha': 0.1}
Best cross-validation score: 0.7202643171806168
Train set accuracy with best parameters: 0.9475770925110132
Test set accuracy with best parameters: 0.7359154929577465
```

```
[[169  94]
 [ 56 249]]
              precision    recall  f1-score   support

   No Stress       0.75      0.64      0.69       263
      Stress       0.73      0.82      0.77       305

    accuracy                           0.74       568
   macro avg       0.74      0.73      0.73       568
weighted avg       0.74      0.74      0.73       568
```

ACCURACY OF ALL MODELS COMPARED:



All models exhibit close accuracy when they have been fine tuned by their hyper parameters as it can be seen above.

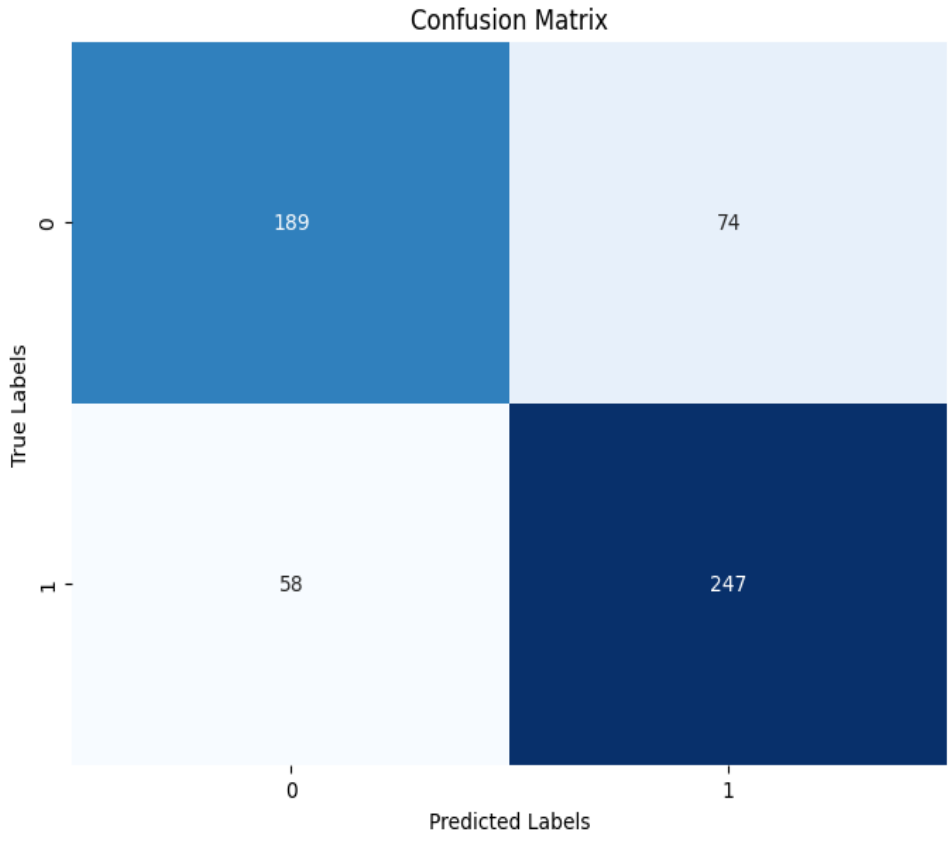
FINAL CLASSIFIER:

Since our individual models i.e Bernoulli Naive Bayes, Logistic Regression, SVM, Random Forest, and Multinomial Naive Bayes exhibited similar accuracy scores, we employed a strategy called ensemble learning. Specifically, we created a Voting Classifier that combines the predictions of these top-performing models. Each model casts a "vote" for the most likely class, and the Voting Classifier makes a final prediction based on the majority vote. This technique proved successful, boosting our overall accuracy to 77%, which surpasses the performance of any of the individual models!

```
[[189  74]
 [ 58 247]]
              precision    recall  f1-score   support

   No Stress       0.77      0.72      0.74       263
      Stress       0.77      0.81      0.79       305

    accuracy                           0.77       568
   macro avg       0.77      0.76      0.77       568
weighted avg       0.77      0.77      0.77       568
```



CHALLENGES FACED:

- **Preprocessing Refinement:** Optimizing the text preprocessing stage proved challenging, particularly in deciding between stemming and lemmatization techniques. Finding the right balance to extract meaningful features while reducing noise was crucial for maximizing model accuracy.
- **Dataset Complexity:** The dataset initially presented challenges due to its numerous columns. Understanding the nature and potential impact of each column on the stress prediction task required careful analysis.
- **Iterative Improvement:** Our initial model achieved an accuracy of 71%. Through meticulous fine-tuning, strategic feature selection, and the implementation of a voting classifier ensemble, we were able to significantly boost the accuracy to 77%.

WAY FORWARD:

We can explore the potential of deep learning models to further enhance our stress detection system. Deep learning architectures like Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, excel at handling sequential data like text. They can capture complex patterns and potentially uncover nuances in language that might be missed by traditional models. However, deep learning often necessitates larger datasets and more computational resources. It's worth investigating if the potential gains outweigh the increased complexity and resource requirements. Additionally, developing a more interactive interface would allow for easier user engagement and real-time stress assessment, enhancing the practical applicability of our system.

CONCLUSION:

Our stress prediction tool, leveraging machine learning and text analysis techniques, offers a promising approach to proactively assess and address stress in individuals. With an accuracy of **77%** through classification and meticulous hyperparameter tuning, we're poised to make meaningful strides in supporting mental well-being.