**Model Flow for Mental Wellness Project**

**1. Data Cleaning**

* Remove Unnecessary Column
* **Null and Duplicate Check:**
  + Remove rows with missing text or class values.
  + Identify and drop duplicate text entries to avoid bias in training.

**2. Class Balance Check**

* **Why?** Imbalanced classes can bias the model towards the majority class.
* **Method:** Check if the number of suicide and non-suicide texts is significantly different.

**3. Feature Engineering**

**Word Count Column**

* Compute the number of words in each text to analyze length distribution.

**Outlier Removal**

* **Why?** Extreme text (Too large or Too less word count) lengths can skew model performance.
* **Method:**
  + Remove texts below the 20th percentile and above the 80th percentile of word count.
* **Action Taken:** Entries with extremely low or high word counts were removed.

**Decreasing Difference in Variance of Word Count Between Two Classes**

* The **variance of word count** between the suicide and non-suicide classes reflects how varied the lengths of the texts are in each class. A **decreasing difference in variance** indicates that the word count distributions across the two classes are becoming similar.
* **Implication in Machine Learning:**
* If the word count variance becomes too similar across classes, this feature alone may not help differentiate between the suicide and non-suicide texts effectively. In such cases, additional features (e.g., text sentiment, specific words, or phrase patterns) become more important.
* **Observation:**
  + Before: Suicide - 36.70, Non-Suicide - 46.61
  + After: Suicide - 20.54, Non-Suicide - 21.81
* Limited length between 50-150

**4. Text Preprocessing**

* Convert text to lowercase.
* Remove punctuation.
* Remove stop words (e.g., “the”, “is”, “in”).
* Tokenization: Split text into words.
* # Lemmatization: Convert words to their base form (e.g., "running" → "run").
* Stemming:

**5. Label Encoding**

* Convert categorical labels into numerical values:
  + **Suicide** → 1
  + **Non-Suicide** → 0

**6. Text Vectorization**

* Convert text into numerical form for model training.
* **Methods Used:**
  + **Unigrams:** Single words as features.
  + **Bigrams (20,000 features):** Two consecutive words to add context.
  + **Trigrams (10,000 features):** Three consecutive words for deeper context.
  + **We used combination of unigram and bigram**

**7. Model Selection: Random Forest**

**8. Model Performance**

* Achieved **88% accuracy** in classifying suicidal vs. non-suicidal posts.

**Advantages:**  
✔ **Effective Data Cleaning & Preprocessing** – Ensures high-quality input.  
✔ **Feature Engineering** – Word count analysis and variance adjustment improve text representation.  
✔ **Balanced Class Distribution** – Prevents model bias.  
✔ **Efficient Vectorization** – Combination of unigrams and bigrams captures context.  
✔ **Good Performance** – 88% accuracy in suicide classification.

**Disadvantages:**  
✖ **Limited Features** – Lacks sentiment analysis, NER, or topic modeling.  
✖ **Outlier Removal Risk** – May discard critical texts.  
✖ **Basic Label Encoding** – Word embeddings (e.g., BERT) could improve results.  
✖ **Random Forest Not Ideal for Text** – LSTMs or Transformers may work better.  
✖ **Limited Explainability** – Deep learning models provide better insights.

**Conclusion:**  
The model is well-structured and performs well, but **adding semantic features** and **using deep learning (e.g., BERT, LSTMs)** could enhance accuracy and context understanding.