A Comparative Study of Classical Models, Feature Spaces, and Reduced-Dimension Techniques for Suicide Detection on Social Media with Geolocation

Team Members:

- Choudari Harshitha Reddy(220010015)
- Prakriti Tripati(220120014)

1. Introduction

Text classification is a central task in natural language processing (NLP), with applications in sentiment analysis, toxicity detection, content moderation, and intent identification. The performance of classical machine learning models depends not only on the algorithms themselves but also on the feature representations used to encode text. Vectorization methods such as TF-IDF, Bag-of-Words (BoW), Chi-Square-selected vocabularies, and latent semantic representations (SVD) offer distinct trade-offs between dimensionality, expressiveness, and computational overhead.

Additionally, dimensionality-reduction methods such as Principal Component Analysis (PCA) and t-SNE can potentially condense information into lower dimensions. However, their suitability for classification tasks remains uncertain.
This project systematically compares **27+ model–feature combinations** and evaluates their classification performance across Accuracy, ROC-AUC, F1 score, and computation time. We aimed to explore how different representations of text affect classification accuracy and what trade-offs exist between model simplicity, explainability, and performance.

Furthermore, we geolocate posts using Named Entity Recognition and create a heatmap of crisis locations to alert health officials and emergency response teams. We also tried experimenting with different platforms like Reddit, X and Instagram.

Our goal is to answer:

1. *Which feature space yields the best performance for classical ML models?*
2. *Does dimensionality reduction (PCA, t-SNE) preserve sufficient information for classification?*
3. *How do boosting models compare to classical linear baselines?*
4. *Can an ensemble improve over the best single model?*
5. *How can we improve geolocation accuracy from text?*
6. *How can we best detect suicidal ideation across different social media platforms?*
7. *Which measure of similarity is best for the Instagram dataset?*
8. *Can we apply the various techniques learnt in class to analyse these datasets?*

2. Dataset & Preprocessing

Data Source

We used the **Suicide_Detection.csv** dataset, a widely used benchmark for text-based suicide ideation classification.
It contains thousands of short posts labeled as either:`suicide` → indicative of suicidal thoughts or self-harm.`non-suicide` → general or neutral posts.

The dataset contains **binary-labelled text samples**, with 7,791 positive and 5,681 negative instances.**Train/Test Split:**80/20 stratified split ensuring balanced class proportions.
Preprocessing included: lowercasing,punctuation removal, tokenization, stopword removal, TF-IDF / BoW vectorization, Chi-Square feature selection (k=1000),Latent semantic compression using SVD-300

For reduced-dimension experiments:

- PCA was applied at 2, 50, and 100 components
- t-SNE was applied as a nonlinear 2D projection(for 1000 samples)

All models were evaluated using consistent train-test splits.

For proof-of-concept testing on current data, we also scraped some Reddit, X, and Instagram posts. We found posts from Reddit were similar to the Suicide_Detection dataset, as they both came from the SuicideWatchers sub-reddit.. Posts from X related to suicide were mostly from the perspective of a third person or a news agency, as people tend not to tweet about their mental health on this platform much. For Instagram posts, we found many to be from users who post only quotes that contain lots of hashtags related to depression and mental health. We modeled the users' liking and commenting on these posts into a dataset. We also added synthetic user IDs, as only an aggregate number of likes and some of the user IDs on a post are displayed.

3. Methodology

We evaluated multiple machine-learning models using four text-representation methods (TF-IDF, BoW, CHI2-1000, SVD-300) and compared their performance across accuracy, F1, ROC-AUC, and runtime. Additionally, we applied dimensionality reduction (PCA, t-SNE) to analyze feature separability and observe how reduced representations affect model behavior.

On the Instagram dataset, we compared user-user similarity across different metrics like Pearson's correlation, cosine similarity, Euclidean similarity, Hamming similarity, and Jaccard similarity. We also experimented with modeling the data as a stream of posts and comparing DGIM and sliding window algorithms to gain insights into how many likes are potentially risky in an hour. On both the Reddit datasets, we applied geolocation and Named Entity Recognition techniques. To improve accuracy, we tried pretrained models, providing a list of 500 cities and blacklisting mispredicted locations.

3.1 Models

A diverse set of classical and modern machine-learning algorithms were evaluated to compare linear, tree-based, neural, and ensemble behaviors on the suicide-detection dataset. Models ranged from simple linear baselines to advanced boosting methods and an ensemble blend combining complementary strengths.KNN was included as an unsupervised reference point. Other models described in class, like bloom filters, DGIM, etc., were also implemented.

- Logistic Regression, Linear Regression (as classifier by thresholding), Neural Network (1 hidden layer), kNN, Decision Tree, Perceptron, XGBoost, LightGBM, Ensemble Blend (Logistic Regression + Neural Network), en_core_web_sm (Spacy), all-MiniLM-L6-v2 (Sentence Transformer), VADER, TextBlob, TF-IDF, BloomFilter, user-user Collaborative Filtering, Reservoir Sampling, Sliding Window, and DGIM.

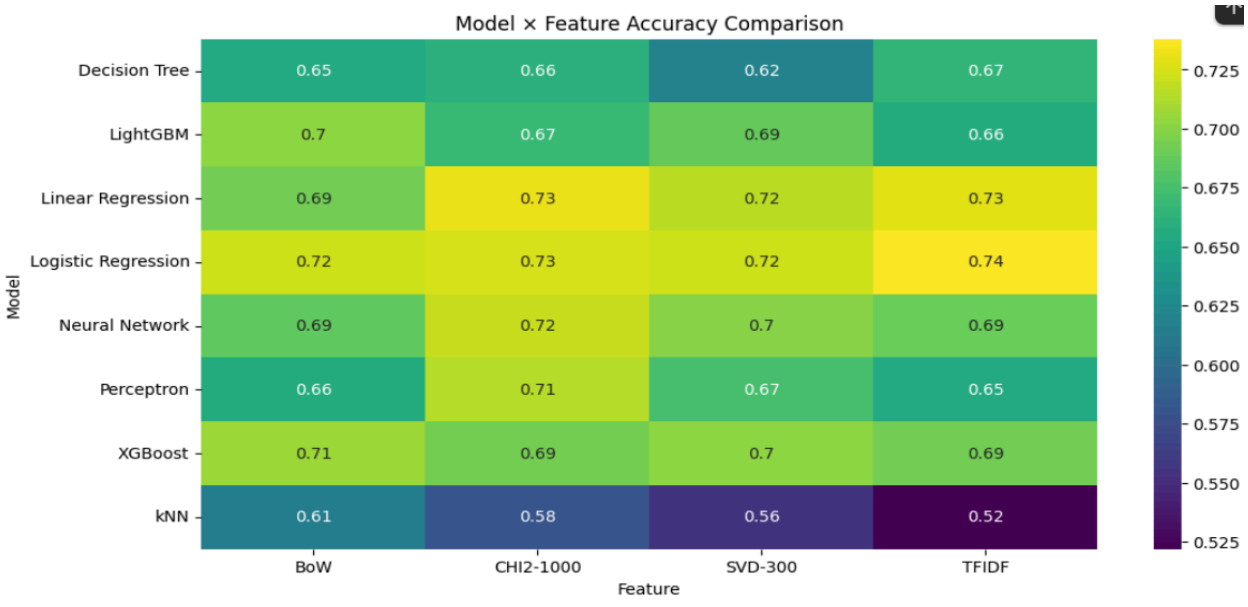
3.2 Feature Representations

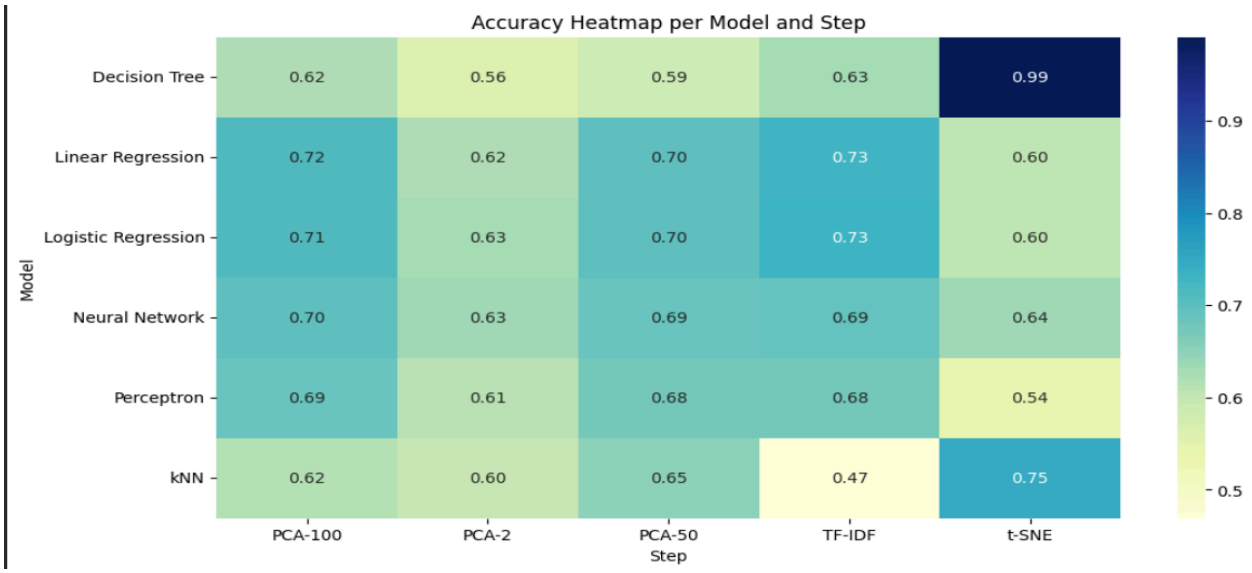
Multiple text-vectorization strategies were tested to study the effect of representation space on model performance. These included sparse lexical features (TF-IDF(high-dimensional sparse), BoW), statistical vocabulary reduction (CHI2-1000), and dense latent semantic embeddings (SVD-300).

3.3 Reduced-Dimension Techniques

Dimensionality reduction was applied to observe how performance changes under compressed or low-rank representations. PCA variants (2, 50, 100 components), t-SNE for nonlinear 2D projection,were used to analyze separability and model robustness. Stratified sampling was used for selecting a subset of the Suicide_Detection.csv dataset for geocoding.

4. Results





-

===== Final Comparison Table =====

	Model	Step	Accuracy	F1	ROC-AUC	Time
0	Logistic Regression	TF-IDF	0.733483	0.772031	0.812758	0.746621
1	Linear Regression	TF-IDF	0.727884	0.763389	0.721540	1.085288
2	Neural Network	TF-IDF	0.692049	0.728529	0.761039	105.482666
3	kNN	TF-IDF	0.469205	0.235484	0.547461	2.256634
4	Decision Tree	TF-IDF	0.631579	0.671984	0.623535	2.886593
5	Perceptron	TF-IDF	0.676372	0.720773	0.667495	1.089128
6	Logistic Regression	PCA-2	0.628219	0.677670	0.619103	0.096837
7	Linear Regression	PCA-2	0.621501	0.667976	0.614279	0.015426
8	Neural Network	PCA-2	0.634938	0.672032	0.632115	1.900866
9	kNN	PCA-2	0.596865	0.649805	0.587378	0.015789
10	Decision Tree	PCA-2	0.564390	0.614470	0.557407	0.029668
11	Perceptron	PCA-2	0.612542	0.590047	0.634795	0.008049
12	Logistic Regression	PCA-50	0.697648	0.741379	0.687719	0.053200
13	Linear Regression	PCA-50	0.698768	0.738581	0.691423	0.078405
14	Neural Network	PCA-50	0.687570	0.728863	0.679971	41.160920
15	kNN	PCA-50	0.650616	0.691700	0.645078	0.120898
16	Decision Tree	PCA-50	0.591265	0.638256	0.584893	0.513644
17	Perceptron	PCA-50	0.675252	0.685466	0.685624	0.016576
18	Logistic Regression	PCA-100	0.709966	0.753568	0.699123	0.127931
19	Linear Regression	PCA-100	0.715566	0.753398	0.708431	0.110654
20	Neural Network	PCA-100	0.698768	0.734975	0.693811	66.014280
21	kNN	PCA-100	0.617021	0.649590	0.616862	0.154925
22	Decision Tree	PCA-100	0.620381	0.668622	0.612281	2.363494
23	Perceptron	PCA-100	0.688690	0.693157	0.702096	0.162276
24	Logistic Regression	t-SNE	0.604000	0.716332	0.556669	0.013433
25	Linear Regression	t-SNE	0.605000	0.717251	0.557537	0.032688
26	Neural Network	t-SNE	0.636000	0.711569	0.610276	2.443043
27	kNN	t-SNE	0.747000	0.789342	0.733392	0.009835
28	Decision Tree	t-SNE	0.991000	0.992126	0.992188	0.008440
29	Perceptron	t-SNE	0.539000	0.481440	0.569019	0.004750

	Model	Feature	Accuracy	ROC-AUC	F1	Time
0	Logistic Regression	TFIDF	0.737962	0.808908	0.776718	0.522139
1	Linear Regression	CHI2-1000	0.732363	0.725439	0.768186	0.053159
2	Linear Regression	TFIDF	0.727884	0.720858	0.764306	0.210839
3	Logistic Regression	CHI2-1000	0.725644	0.804086	0.767773	0.079882
4	Logistic Regression	BoW	0.722284	0.791226	0.751503	0.272051
5	Logistic Regression	SVD-300	0.721165	0.800608	0.761722	0.265190
6	Neural Network	CHI2-1000	0.720045	0.779101	0.750996	29.349941
7	Linear Regression	SVD-300	0.716685	0.710429	0.753171	0.081439
8	Perceptron	CHI2-1000	0.714446	0.709162	0.749755	0.018482
9	XGBoost	BoW	0.706607	0.784865	0.728216	4.541023
10	LightGBM	BoW	0.702128	0.760003	0.735586	5.247452
11	XGBoost	SVD-300	0.702128	0.767772	0.748582	43.045976
12	Neural Network	SVD-300	0.699888	0.771096	0.750929	54.378021
13	XGBoost	CHI2-1000	0.693169	0.773946	0.722672	3.293076
14	XGBoost	TFIDF	0.692049	0.768095	0.723618	11.104901
15	Linear Regression	BoW	0.692049	0.693080	0.719101	0.124604
16	Neural Network	TFIDF	0.688690	0.748061	0.723658	98.570132
17	LightGBM	SVD-300	0.687570	0.756030	0.739496	86.193709
18	Neural Network	BoW	0.686450	0.747676	0.722222	65.027089
19	Perceptron	SVD-300	0.674132	0.698635	0.653159	0.051617
20	LightGBM	CHI2-1000	0.667413	0.732946	0.705065	8.505095
21	Decision Tree	TFIDF	0.665174	0.655843	0.709427	1.337264
22	Decision Tree	CHI2-1000	0.660694	0.653037	0.691131	0.353906
23	LightGBM	TFIDF	0.656215	0.735744	0.701071	16.009520
24	Perceptron	BoW	0.656215	0.630166	0.729038	0.087348
25	Decision Tree	BoW	0.653975	0.650600	0.704871	2.063617
26	Perceptron	TFIDF	0.653975	0.638791	0.710945	0.023499
27	Decision Tree	SVD-300	0.620381	0.613881	0.662014	6.038737
28	kNN	BoW	0.613662	0.674054	0.627832	0.199681
29	kNN	CHI2-1000	0.580067	0.607879	0.660633	0.194925
30	kNN	SVD-300	0.555431	0.602680	0.505604	0.438989
31	kNN	TFIDF	0.521837	0.566074	0.500585	0.232646

- **TF-IDF + Logistic Regression is the strongest single model**, with the highest ROC-AUC and F1 . **CHI2-1000** performs surprisingly close, showing that reducing features still preserves important signals.**TF-IDF and CHI2-1000 give the strongest overall performance**, especially for Logistic Regression and Linear models

- Boosting performs well, but still **does not outperform Logistic Regression + TF-IDF**. This indicates that the dataset is linearly separable in the TF-IDF space.
- Dimensionality reduction **hurts performance drastically**.**PCA retains reasonable structure**, but performance clearly drops as dimensionality is reduced—showing some information loss.
- PCA-50 is the best reduced variant on the basis of dim-acc, but still far below SVD or CHI2.
- Ensemble Blend (Logistic Regression, Neural Network) : Accuracy=0.701 F1=0.740 ROC=0.795. The ensemble improves recall and F1 but **does NOT outperform TF-IDF Logistic Regression alone**.This further supports that linear models already capture most decision boundaries.
- **t-SNE performs worst for classification**, confirming it is not suitable as a feature space for supervised models.
- **SVD-300 provides competitive accuracy with much lower dimensionality**, showing good information retention.
- **Bag-of-Words works moderately well but is consistently weaker than TF-IDF and CHI2**, especially for neural networks and tree models.
- **kNN and Decision Trees perform poorly across all features**, confirming they are not suitable for sparse text representations.

Fig. 1:

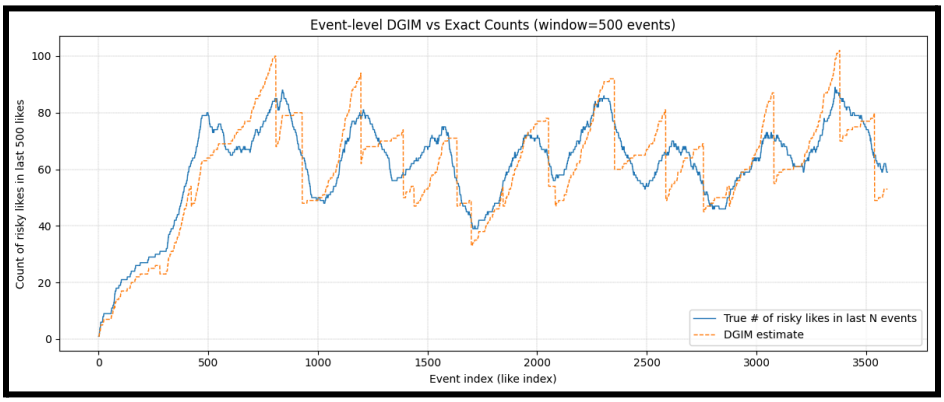


Fig. 2:

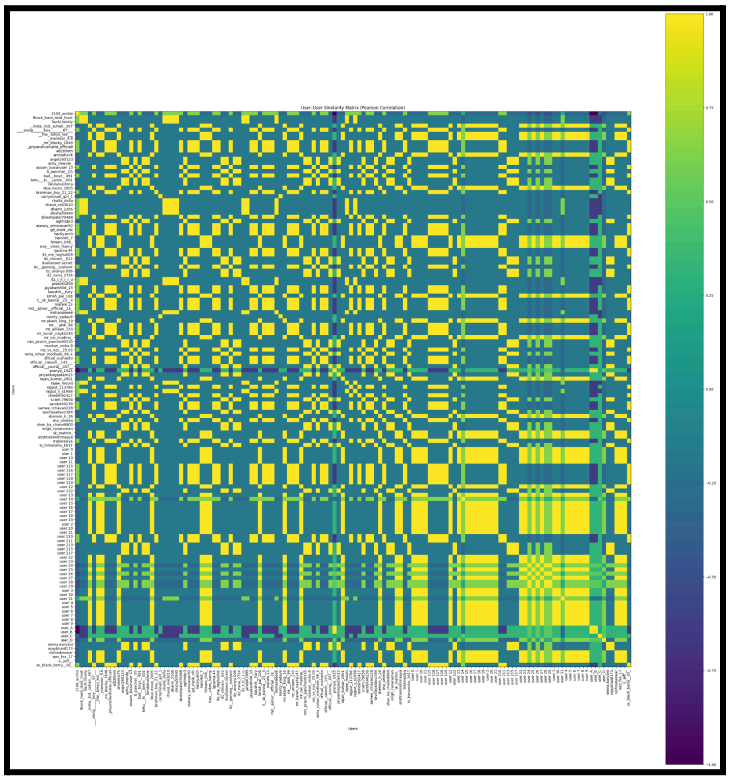
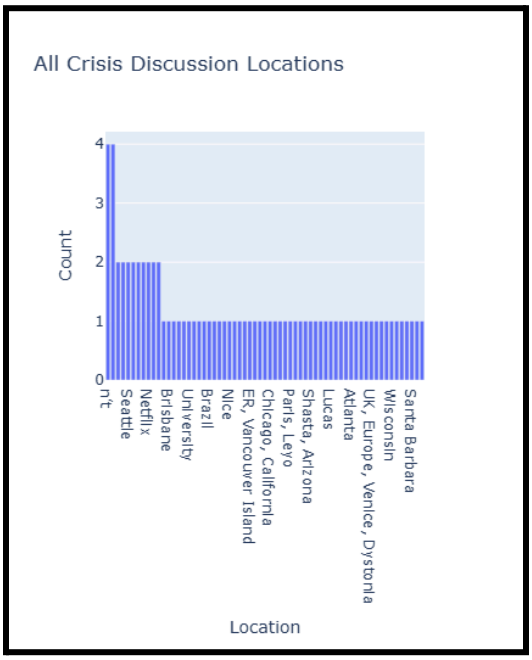
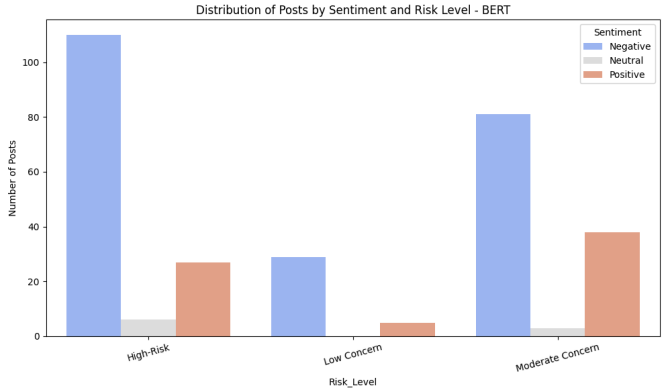
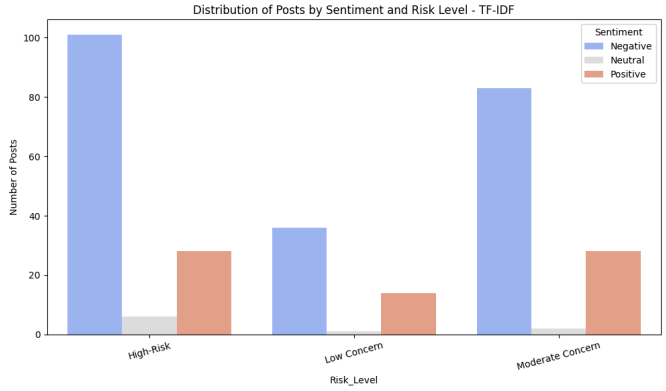


Fig. 3:

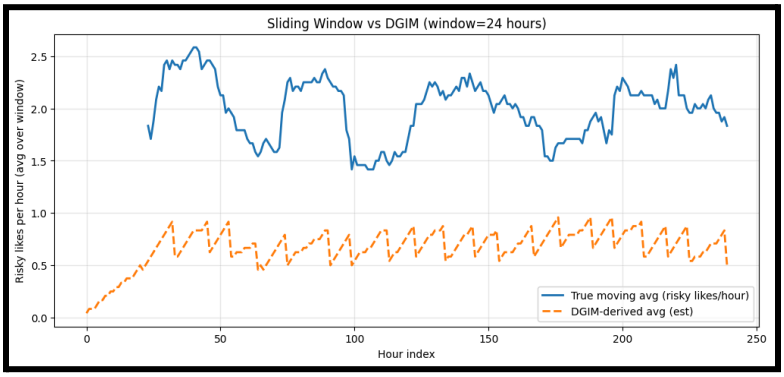


- For Reddit posts scraped from the API, BERT and TF-IDF produce distributions that are similar in shape, as shown below.





- Geolocation on the Suicide_Detection.csv dataset gives the interactive .html map at: https://prakriti16.github.io/SDM_Project/ . Figure 3 above shows a chart of the locations.
- Figure 2 above shows Pearson's coefficient user-user similarity matrix for the Instagram dataset. Similar plots for other similarity measures are attached in the appendix. These have a similar distribution to each other but vary in magnitude and intensity.
- The Bloom filter gave lower false positives when its size was comparable to the number of items. Reservoir sampling outputs a different set of samples at each run.
- As shown in Figure 1 above, when the DGIM algorithm is applied to every event in the data stream, its performance is very close to the true counts given by the Sliding Window approach. Here, each bit in the data stream represents whether the user liked a suicide related risky post at that timestep. However, if we apply it as an aggregate, we get worse results as shown:



5. Discussion

- **Why TF-IDF Wins?** TF-IDF creates a high-resolution sparse representation of term importance, ideal for linear classifiers. The separability and high ROC-AUC indicate strong signal concentration.
- **Why BoW Comes Close?** BoW captures raw frequency effectively. Despite its simplicity, Logistic Regression performs strongly on BoW (0.769 accuracy).
- **Why CHI2-1000 Works Surprisingly Well ?** The CHI2-reduced vocabulary filters uninformative terms, improving generalization.
  Its performance nearly matches TF-IDF.
- **SVD-300** Dense semantic representations help neural networks but reduce linear model performance slightly.

- **Why PCA/t-SNE Fail?** PCA and t-SNE reduce dimensionality too aggressively, discarding discriminative features required for classification.
- **Why is the Geolocation accuracy sometimes low?** If we try to find locations based on similarities to a list of places or to other locations and a small binary classifier model or Named Entity Recognition, it may lead to names of people and companies being confused with locations, as they all fall under the category of Proper Nouns.
- **Why Event-Level DGIM Performs So Well?** When DGIM is applied per event (each like = a bit), it closely matches the true Sliding Window count (Figure 1). Event-level DGIM captures spikes and variations in risky likes with high temporal resolution. Its logarithmic memory structure maintains accuracy even on long streams. This makes DGIM highly effective for real-time monitoring of suicide-related user activity.

6. Conclusion and Future Work

This report is a comprehensive analysis across: 5+ models, 4 major feature spaces, 3 PCA compressions, t-SNE, 2 boosting frameworks, 1 ensemble method

Best Overall Model : Logistic Regression + TF-IDF

DGIM algorithm per event is best for modelling data streams in real time. Geolocation accuracy can be improved in the future by using pretrained transformer models that can differentiate between names of places and names of people or companies based on the context. But as a starting point, using blacklists and similarity measures also works.

Major Findings

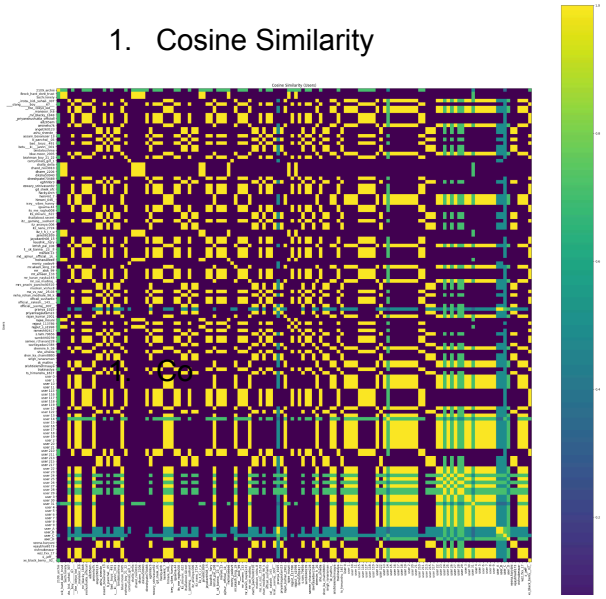
- High-dimensional sparse features outperform dense compressed representations.
- PCA and t-SNE drastically degrade classifier performance.
- LightGBM and XGBoost are strong but still weaker than linear baselines.
- Ensemble blending improves stability but not peak performance.
- Logistic Regression remains the most robust and interpretable model for text classification.
- DGIM is useful for modelling posts as data streams.
- Geolocation accuracy can be improved by using blacklists and similarity detection methods.

7. References:

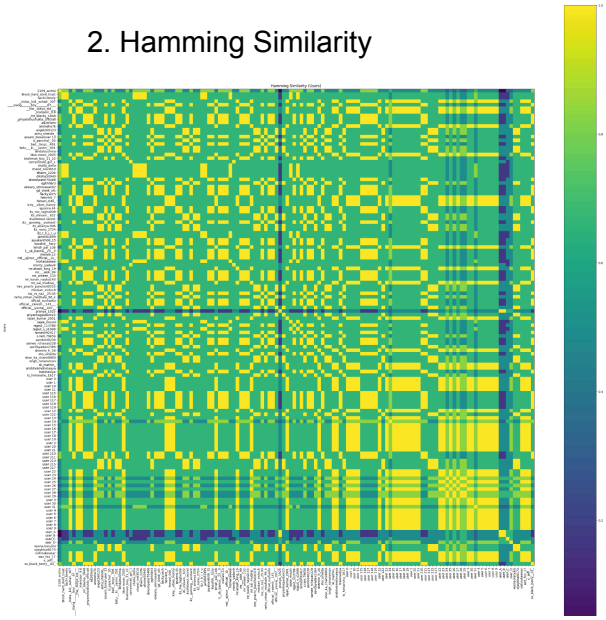
1. Knowledge-aware Assessment of Severity of Suicide Risk for Early Intervention | The World Wide Web Conference
2. midas-research/sismo-wsdm: Code release for "Towards Ordinal Suicide Ideation Detection on Social Media", WSDM 2021.
3. Suicide ideation detection from online social media: A multi-modal feature based technique - ScienceDirect
4. Predicting acute suicidal ideation on Instagram using ensemble machine learning models - ScienceDirect

8. Appendix:

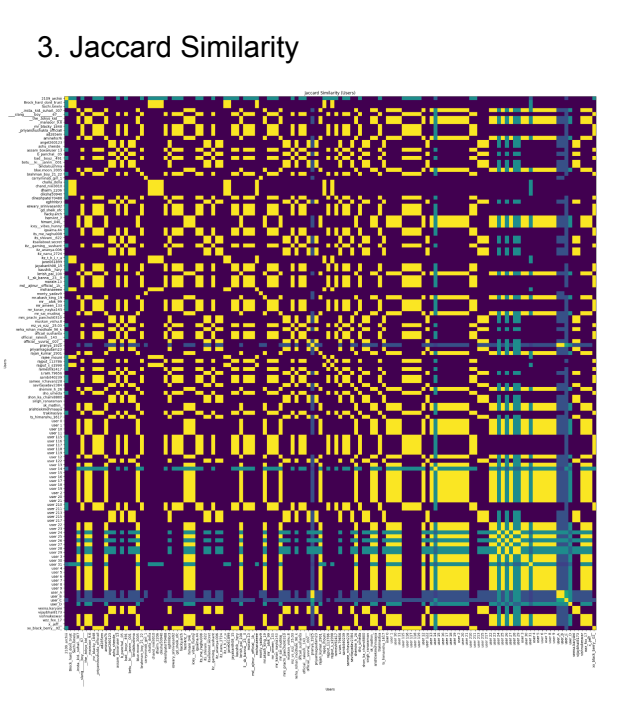
1. Cosine Similarity



2. Hamming Similarity



3. Jaccard Similarity



4. Euclidean Similarity