IXA Internship Report: Early Risk Detection

Luis Antonio VASQUEZ REINA

July 12, 2023

1 Introduction

The *early risk detection task* involves identifying and predicting potential risks or hazards at an early stage before they fully manifest or escalate ([PMRLC21]). It typically applies to various domains, including finance, cybersecurity, healthcare, and natural disasters, among others.

In this task, the goal is to develop models or systems that can analyze relevant data sources, such as historical records, real-time data streams, or sensor readings, to detect patterns or indicators that may signify an increased risk or the emergence of an adverse event. By identifying these early warning signs, proactive measures can be taken to mitigate or prevent the potential negative impact.

The ultimate objective of early risk detection is to enable timely decision-making, intervention, or precautionary measures to minimize the impact of adverse events and safeguard individuals, organizations, or systems from potential harm. It plays a crucial role in risk management, operational efficiency, and ensuring the safety and well-being of individuals and assets.

In this project, we develop an approach to early risk detection based on the analysis of the gradual change in the content of documents produced by a user. That is, we leverage *topic modelling* to extract the main content of a user's documents, and we analyze the evolution of their topic to classify a user as *at risk* or *not at risk*. Note that for risk detection, false negatives are to be avoided, as it is more dangerous to ignore people at risk than to give attention to people without risk.

2 Our Approach

Our main idea is that the evolution of the topics of the documents from a user may give an insight into the condition of the user. To implement this, we leverage the following techniques.

2.1 Topic modelling

BERTopic ([Gro22]) is a topic modelling approach based on sentence embeddings. It clusters documents using cosine similarity as a measure of distance. After clustering documents into topics, BERTopic can take a new document and produce a probability distribution for the topics, that is, the probabilities that the document belongs to each topic (Figure 1).

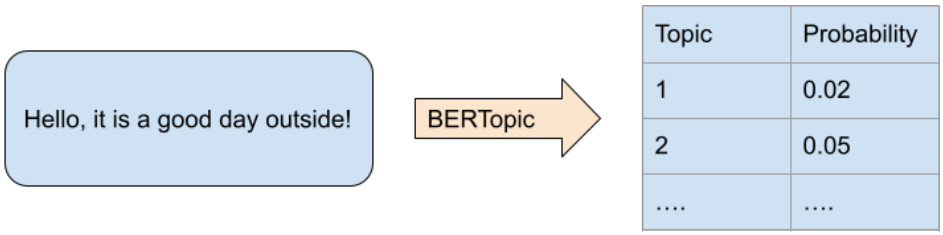

Figure 1: BERTopic

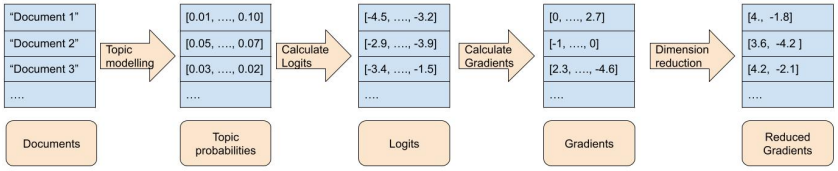

Figure 2: Steps for gradient calculation

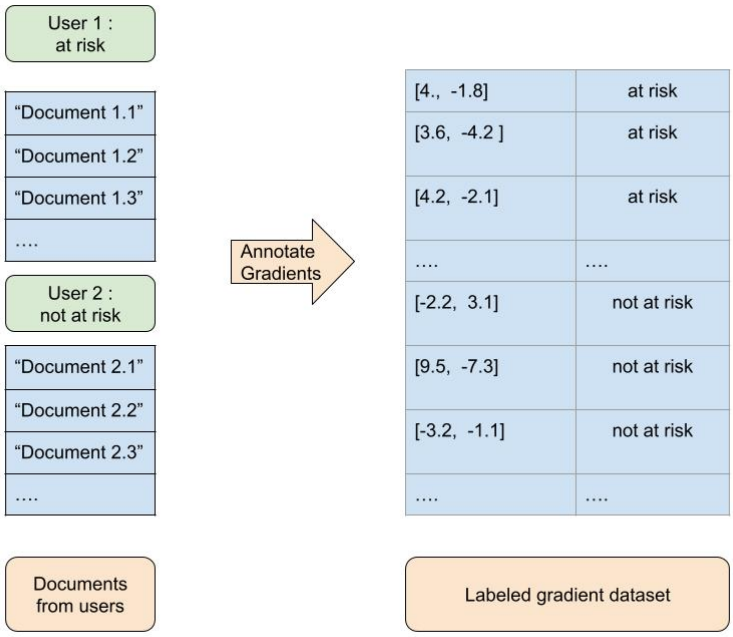

Figure 3: Labeling gradients from user documents

2.2 Gradient Calculation

The probabilities produced by BERTopic usually are of small magnitude, specially when clustering into a high number of topics. For this reason, we first calculate the *logits* (simply the natural logarithm of the probabilities), and calculate the gradients as the successive differences of logits. After inspection, we noticed that these gradients are very sparse, thus, we used Principal Component Analysis (PCA) for *dimensionality reduction* (Figure 2).

2.3 Gradient Classifier

Taking the (dimension-reduced) gradients as features, and an annotated dataset for users at risk, we annotate the gradients, by assigning the user label to all the gradients from the user documents (Figure 3). We then train a *logistic regression* classifier using this data. The main idea behind this step is to teach to the classifier "this gradient corresponds to risky behavior".

2.4 User Classification Algorithm

This is the algorithm for classifying users based on their documents:

- The documents must be given in chronological order.
- Using the topic model, we calculate the topic probabilities for each document.
- We calculate the logits.
  - To avoid $\log(0)$, we add $\varepsilon = 1e-4$ to the probabilities.
- We calculate the gradients as the difference of the logits of consecutive documents.
  - Given that calculating the gradients reduces the list length by one, we add a dummy document (empty string) at the beginning of the document list.
- We apply PCA dimensionality reduction on the gradients.
- Using the (reduced) gradient classifier, we go through the gradients:
  - On a gradient, calculate the probabilities that it corresponds to the positive or the negative label
  - If one of the probabilities is above a predefined threshold (for example, 80%), assign the corresponding label to the user, and *stop* the classification.
    - This is done because we focus on *early* detection.
  - If none of the probabilities is above the predefined threshold, move on to the next gradient.
  - Iterate until a label is assigned.
  - If no label is assigned, label the user as at risk (assign the positive label)
    - This is done to minimize false negatives.
  - Then, we register the number of documents needed to make a decision (that is, assigning a probability above the threshold, or reaching the final document).
- Finally, we evaluate the performance of our algorithm using the $F_{latency}$ metric for early risk detection.

3 Dataset

We use the Sentiment140 [GBH09], which we downloaded from Kaggle[1]. The Sentiment140 dataset is a widely-used dataset in the field of Natural Language Processing (NLP) and sentiment analysis. It consists of a collection of tweets from a multitude of users at different dates, and are labeled with sentiment polarity. The dataset was compiled from Twitter and has been widely used for sentiment analysis tasks, including sentiment classification and sentiment intensity prediction.

The sentiment labels in the dataset indicate whether a particular tweet expresses a positive sentiment or a negative sentiment. Each tweet is labeled with either "0" (indicating a negative sentiment) or "4" (indicating a positive sentiment). The size of the dataset is exactly 1.6 million tweets, and it is balanced in both categories.

4 Our Pipeline

4.1 Preprocessing

After inspection, the first step in preprocessing the Sentiment140 dataset was to eliminate extremely short tweets (less than 5 tokens) and tweets belonging to bots. Afterward, we only kept those users with at least 5 tokens. Given that the documents come from Twitter, we then proceed to perform

[1] https://www.kaggle.com/datasets/kazanova/sentiment140

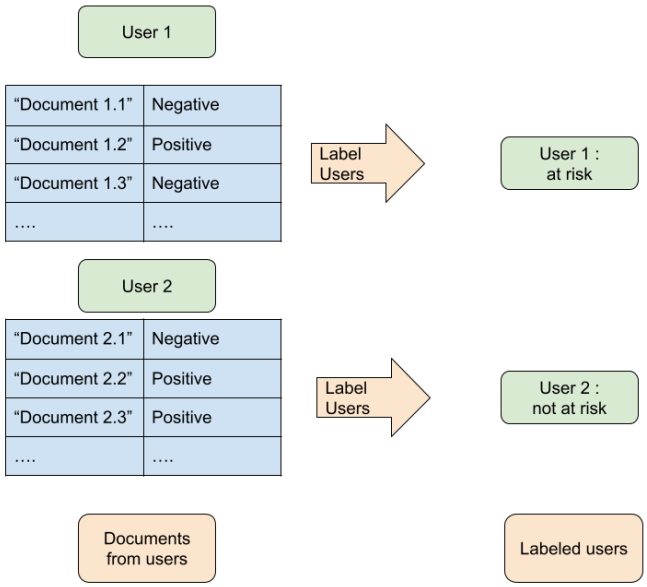

Figure 4: Strategy for labelling users

Topic	keywords
1	dog, animals, puppy, pets, paws, zoo, pet, cagestanks, pupay, pony
2	germany, german, berlin, germans, thuringia, bavaria, jena, na**s, na**
3	uk, england, come, wirral, liverpool, mancherster, finsbury, 03, rpattz, stonehenge

Table 1: Some topics obtaine by BERTopic

basic cleaning of the text, by deleting emojis, URLs, user mentions ("@user"), extra white spaces and converting to lower case[2].

Since our task is early risk detection of *users* based on their documents, and the Sentiment140 datasets only annotates documents, we designed the following strategy for creating a dataset of users at risk: Given a user and their documents in chronological order, the user is labeled as *at risk* if the majority (at least two thirds) of their documents have a negative sentiment. Otherwise, the user is labeled as *not at risk* (Figure 4).

Our strategy produced a dataset of 59794 users with risk labels and their documents in chronological order. This is the type of dataset we described in Section 2.3. Finally, we make an 80-20 train and test split, giving us 47835 users for the training dataset, and 11959 for the test dataset, however, due to computational limitations, we had to downsample the training dataset to 5% of its original size, that is 2391 users.

4.2 Training BERTopic

BERTopic [Gro22] is a BERT-based topic modelling tool which, after producing BERT for sentence embeddings, merges documents into clusters by cosine similarity. One of the hyperparameters for BERTopic is the minimum number of different topics. For each of the training documents, BERTopic produces an array of probabilities of belonging to each of the different topics. Finally, a trained BERTopic instance can be used to get probabilities for new documents.

Our next step was to train BERTopic with the documents from the users from our training data, requiring at least 100 different topics. This step is the most computationally demanding from our pipeline. After training, BERTopic found 204 different topics. Some sample topics are in Table 1.

[2]It is common for tweets to be written in all capital letters

Metric	Value
F_1	0.36
speed	0.98
$F_{latency}$	0.35

Table 2: Metrics on test dataset

4.3 Processing the gradients

We proceed to create a dataset of gradients corresponding to risky behavior as described in sections 2.2 and 2.3. As a summary, for user u, logits are calculated using the following formula, where $\varepsilon = 1e-4$, and T is the set of topics:

$$logit_{u,i} = \ln\left(p_{u,i} + \varepsilon\right) \text{ for all } i \in T$$

Then, given that the documents were in chronological order, the gradients are calculated as successive differences. Since this reduces the number of gradients by one, we include a dummy document, the empty string, at the beginning of the document list. Since this produces very sparse vectors, we use PCA dimension reduction into 20 components.

$$grad_{u,i} = PCA(logit_{u,i} - logit_{u,i-1}) \text{ for all } i \in T$$

Afterward, the dimension-reduced gradients inherit the risk label from their corresponding user.

$$\text{Gradient Dataset } = \{(grad_{u,i}, label_u) \text{ for } i \in T, u \in \text{Users}\}$$

Finally, as explained in section 2.3, we train a logistic regression classifier on this dataset.

5 Evaluation and Discussion

Traditional classification methods are not designed to take into account the delay of the system in making a decision. This is crucial for early risk detection. Apart from the usual measure of accuracy, precision, recall, and F_1 scores, we evaluate our pipeline using the $F_{latency}$ metric for early risk detection, as defined in [PMRLC21].

The algorithm described in section 2.4 registers the number of documents it takes to make a decision. [PMRLC21] defines the *penalty* of taking a decision after k documents as:

$$\text{penalty}(k) = -1 + \frac{2}{1 * \exp\left(-p(k-1)\right)}$$

Where the p parameter controls how quickly the penalty get worse, and which the authors experimentally set at $p = 0.0078$. Notice that $\text{penalty}(1) = 0$, that is, a system that takes a decision about the user after watching a single document gets no penalty[3].

For early risk detection, the focus is on how quickly the system detects *true positives*, thus the *speed* of a system is defined as

$$speed = (1 - \text{median}\{\text{penalty}(k_u), u \in \text{Users}, \text{decision}(u) = 1 \text{ and } \text{label}(u) = 1\}$$

where k_u is the number of documents necessary to take a decision about the user u, $\text{decision}(u)$ is the decision of the system, and $\text{label}(u)1 =$ means the user is at risk. For our system,

The $F_{latency}$ score is then defined as

$$F_{latency} = F_1 * speed$$

Our system consists of applying the gradient classifier with the algorithm described in section 2.4, using a decision threshold of 80%. After evaluating our system on the test data, we obtained the results in Table 2.

[3] Additionally, the definition assumes 1-indexing for the documents

While the classifier demonstrates agility in making prompt decisions for as true positives, it regrettably falls short when it comes to achieving a consistently high level of overall decision accuracy. That is, the classifier's proficiency in swiftly identifying positive cases is overshadowed by its inherent limitations in maintaining a consistently reliable decision-making process across various scenarios.

6 Future work

A fundamental aspect for future work would be to use make effective use of the entirety of the training data that is currently at our disposal, a task that has been impeded by the constraints imposed by computational limitations, which forced us to downsample the training data to 5% of its size. Additionally, there are several hyperparameters that could be tweaked to explore better configurations:

- The minimum number of topics for BERTopic (currently 100)
- The number of component for PCA dimension reduction of the gradients (currently 20)
- The probability threshold to take a classification decision (currently 80%)
- The p parameter for calculating the penalties for $F_{latency}$ (currently 0.0078)
- The type of probabilistic classifier used for the gradients (currently, logistic regression)

References

[GBH09] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009, 2009.

[Gro22] Maarten Grootendorst. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*, 2022.

[PMRLC21] Javier Parapar, Patricia Martín-Rodilla, David E Losada, and Fabio Crestani. Overview of erisk at clef 2021: Early risk prediction on the internet (extended overview). *CLEF (Working Notes)*, pages 864–887, 2021.