# Actuarial Applications of Natural Language Processing Using Transformers

## Case Studies for Using Text Features in an Actuarial Context

Insurance Data Science Conference, London

16th June 2023

Dr. Jürg Schelldorfer, Actuary SAA

Lead Actuarial Data Scientist, Swiss Re

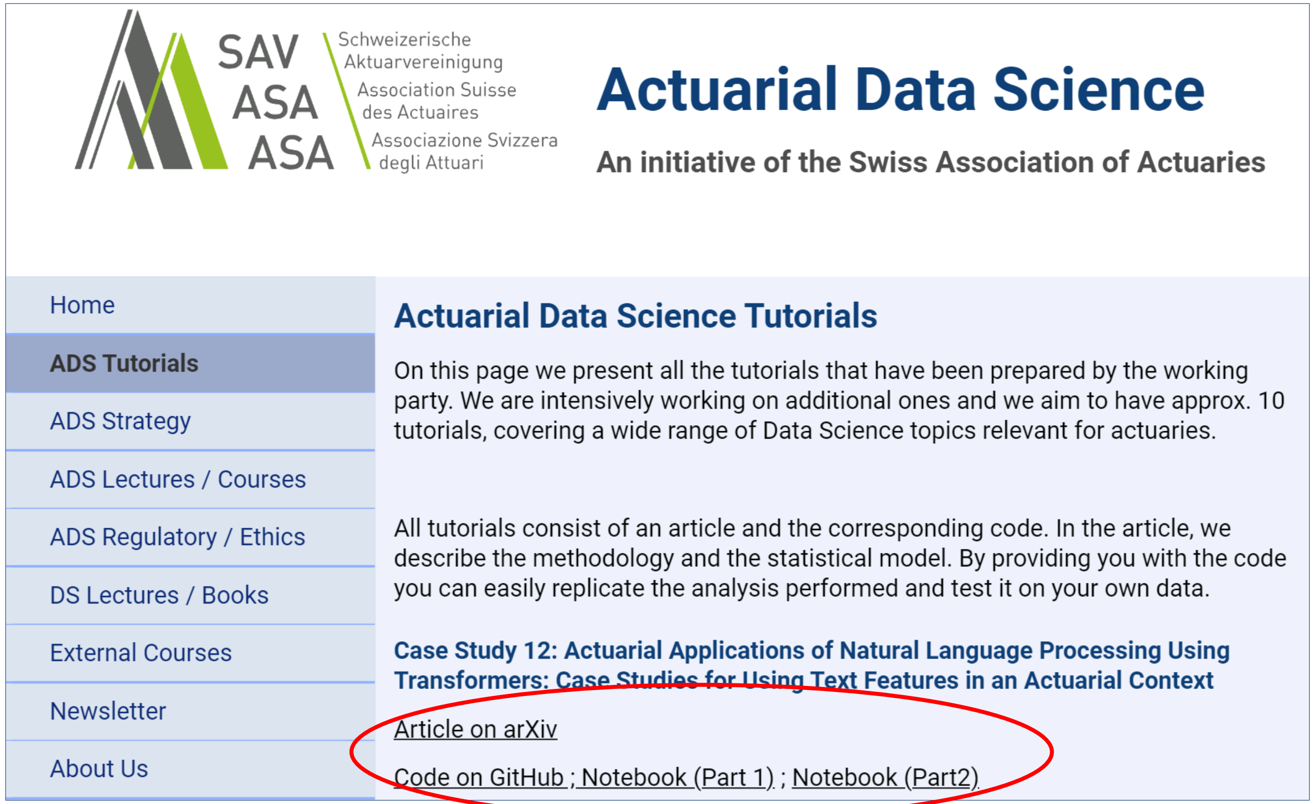
Former chair of the «Data Science» working group, Swiss Association of Actuaries (SAA)

# Disclaimer

The opinions expressed in this presentation are those of the author only. They are inspired by the work that the author is doing for both Swiss Re and the SAA, but they do not necessarily reflect any official view of either Swiss Re or the SAA.

# Tutorial

www.actuarialdatascience.org



SAV ASA ASA
Schweizerische Aktuarvereinigung
Association Suisse des Actuaires
Associazione Svizzera degli Attuari

Actuarial Data Science

An initiative of the Swiss Association of Actuaries

- Home
- ADS Tutorials
- ADS Strategy
- ADS Lectures / Courses
- ADS Regulatory / Ethics
- DS Lectures / Books
- External Courses
- Newsletter
- About Us

Actuarial Data Science Tutorials

On this page we present all the tutorials that have been prepared by the working party. We are intensively working on additional ones and we aim to have approx. 10 tutorials, covering a wide range of Data Science topics relevant for actuaries.

All tutorials consist of an article and the corresponding code. In the article, we describe the methodology and the statistical model. By providing you with the code you can easily replicate the analysis performed and test it on your own data.

Case Study 12: Actuarial Applications of Natural Language Processing Using Transformers: Case Studies for Using Text Features in an Actuarial Context

Article on arXiv

Code on GitHub ; Notebook (Part 1) ; Notebook (Part2)

https://arxiv.org/abs/2206.02014

Actuarial Applications of Natural Language Processing Using Transformers

Case Studies for Using Text Features in an Actuarial Context

Andreas Troxler * Jürg Schelldorfer **

v1, 3 June 2022

Abstract

This tutorial demonstrates workflows to incorporate text data into actuarial classification and regression tasks. The main focus is on methods employing transformer-based models. A dataset of car accident descriptions with an average length of 400 words, available in English and German, and a dataset with short property insurance claims descriptions are used to demonstrate these techniques. The case studies tackle challenges related to a multi-lingual setting and long input sequences. They also show ways to interpret model output, to assess and improve model performance, by fine-tuning the models to the domain of application or to a specific prediction task. Finally, the tutorial provides practical approaches to handle classification tasks in situations with no or only few labeled data. The results achieved by using the language-understanding skills of off-the-shelf natural language processing (NLP) models with only minimal pre-processing and fine-tuning clearly demonstrate the power of transfer learning for practical applications.

**Keywords.** Natural language processing, NLP, transformer, multi-lingual models, domain-specific fine-tuning, integrated gradients, extractive question answering, zero-shot classification, topic modeling.

# Table of Content

# Data

# Data (1/4)

## Wisconsin Local Government Property Insurance Fund (LGPIF)

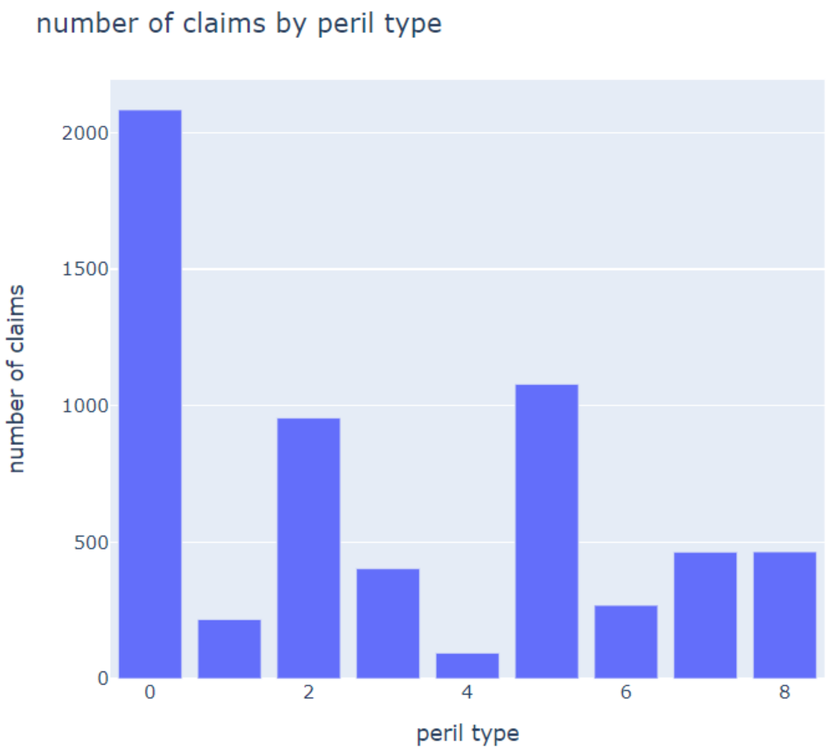
- The data consists of **6’030 records** (4’991 in the training set, 1’039 in the test set) which include a claim amount, a short English claim description and a hazard type with 9 different levels: Fire, Lightning, Hail, Wind, WaterW (weather related water claims), WaterNW (other weather claims), Vehicle, Vandalism and Misc (any other).
- The following exhibit shows an example

| row | Vandalism | Fire | Lightning | Wind | Hail | Vehicle | WaterNW | WaterW | Misc | Loss | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 6838.87 | lightning damage |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2085 | lightning damage at Comm. Center |
| 6 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 8775 | surveillance equipment stolen |
| 7 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 34610.27 | wind blew stack off and damaged roof |
| 9 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 9711.28 | forklift hit building damaging wall and door frame |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1942.67 | water damage at courthouse |
| 30 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 3469.79 | light pole damaged |

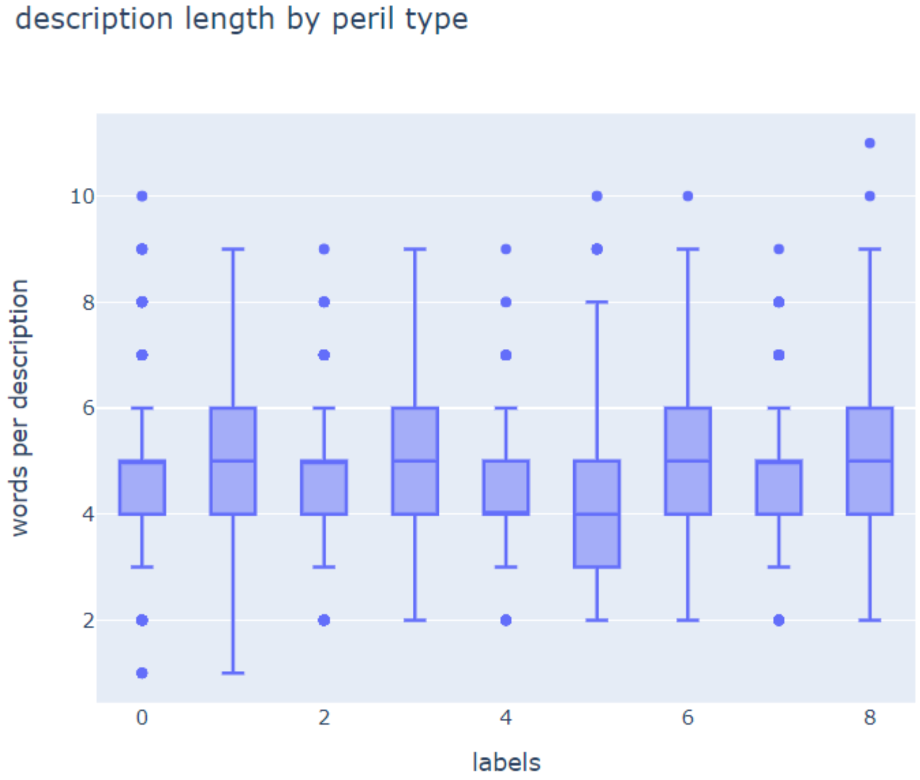
https://github.com/OpenActTexts/Loss-Data-Analytics/tree/master/Data

# Data (2/4)

## Wisconsin Local Government Property Insurance Fund (LGPIF)



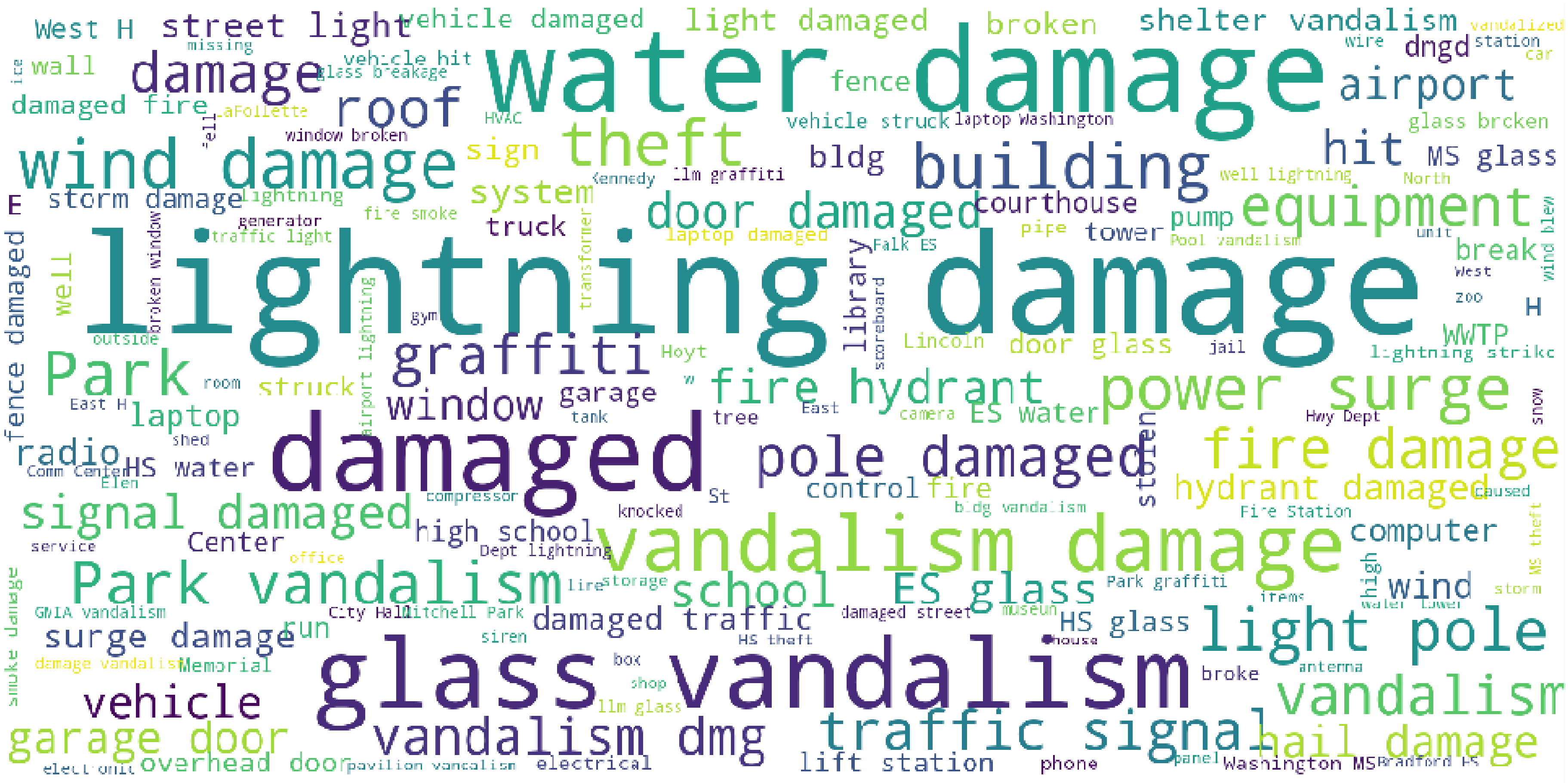

| | peril | train | valid |
|---|---|---|---|
| 0 | Vandalism | 1774 | 310 |
| 1 | Fire | 171 | 46 |
| 2 | Lightning | 832 | 123 |
| 3 | Wind | 296 | 107 |
| 4 | Hail | 76 | 18 |
| 5 | Vehicle | 852 | 227 |
| 6 | WaterNW | 202 | 67 |
| 7 | WaterW | 426 | 38 |
| 8 | Misc | 362 | 103 |
| 9 | Total | 4991 | 1039 |




Overall number of words by claim description:
min 1, average 5, max 11

# Data (3/4)

## Wisconsin Local Government Property Insurance Fund (LGPIF)

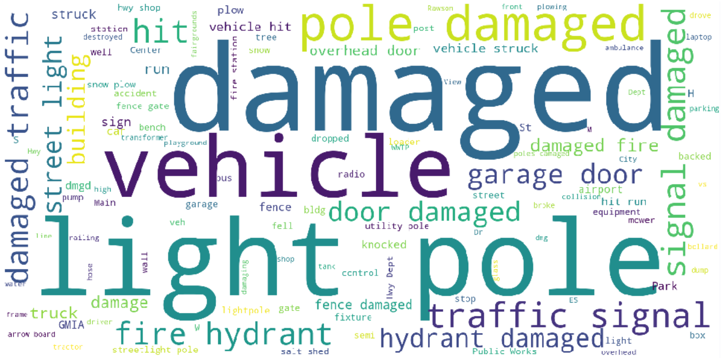


# Data (4/4)

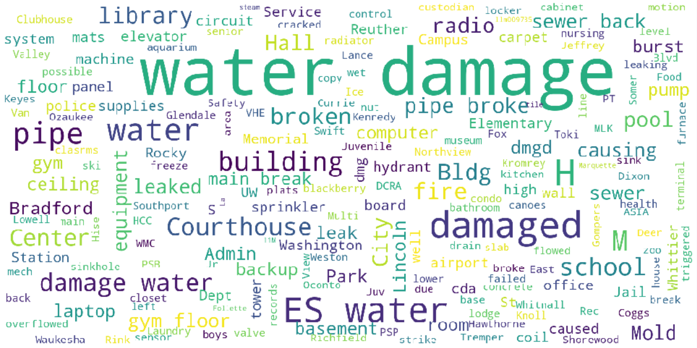
## Wisconsin Local Government Property Insurance Fund (LGPIF)

Vandalism

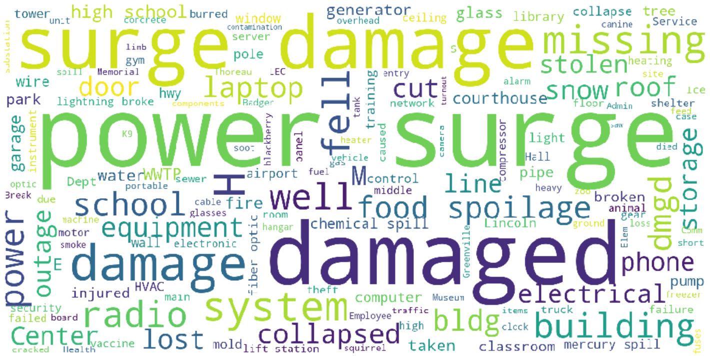

Vehicle




WaterNW




Misc



# Framing the Business and Analytics Problem

- Business Problem: Classify the claims into the 8 categories based on the claims description.
- Analytics Problem: short property insurance claim description which we aim to classify by peril type.

- Classify by peril type in a supervised setting
  - To warm up, we apply supervised learning techniques you have learned in Part I to the dataset of this Part II.
- Zero-shot classification
  - This technique assigns each text sample to one element of a pre-defined list of candidate expressions. This allows classification without any task-specific training and without using the labels. This fully unsupervised approach is useful in situations with no labels.
- Unsupervised classification using similarity
  - This technique encodes each *input sentence* and each *candidate expression* into en embedding vector. Then, pairwise similarity scores between each input sequence and each candiate expression are calculated. The candidate expression with the highest similarity score is selected. This fully unsupervised approach is useful in situations with no labels.
- Unsupervised topic modeling by clustering of document embeddings
  - This approach extracts clusters of similar text samples and proposes verbal representations of these clusters. The labels are not required, but may be used in the process if available. This technique does not require prior knowledge of candidate expressions.

# Classify by peril type in a supervised setting

# High-level approach

| Label (Y) | Description (X) |
|---|---|
| Lightning | lightning damage |
| Vandalism | surveillance equipment stolen |
| Wind | wind blew stack off and damaged roof |

How to fit a supervised model, when the feature space are words?

➔ First idea: Encode the words with one-hot-encoding like categorical features. This results in a very high-dimensional, sparse matrix X.

| Y | Lignhtning | Damage | Center | Surveillance | Equipment | stolen | … |
|---|---|---|---|---|---|---|---|
| Lightning | 1 | 1 | 0 | 0 | 0 | 0 | … |
| Vandalism | 0 | 0 | 0 | 1 | 1 | 1 | … |
| Wind | 0 | 1 | 0 | 0 | 0 | 0 | … |

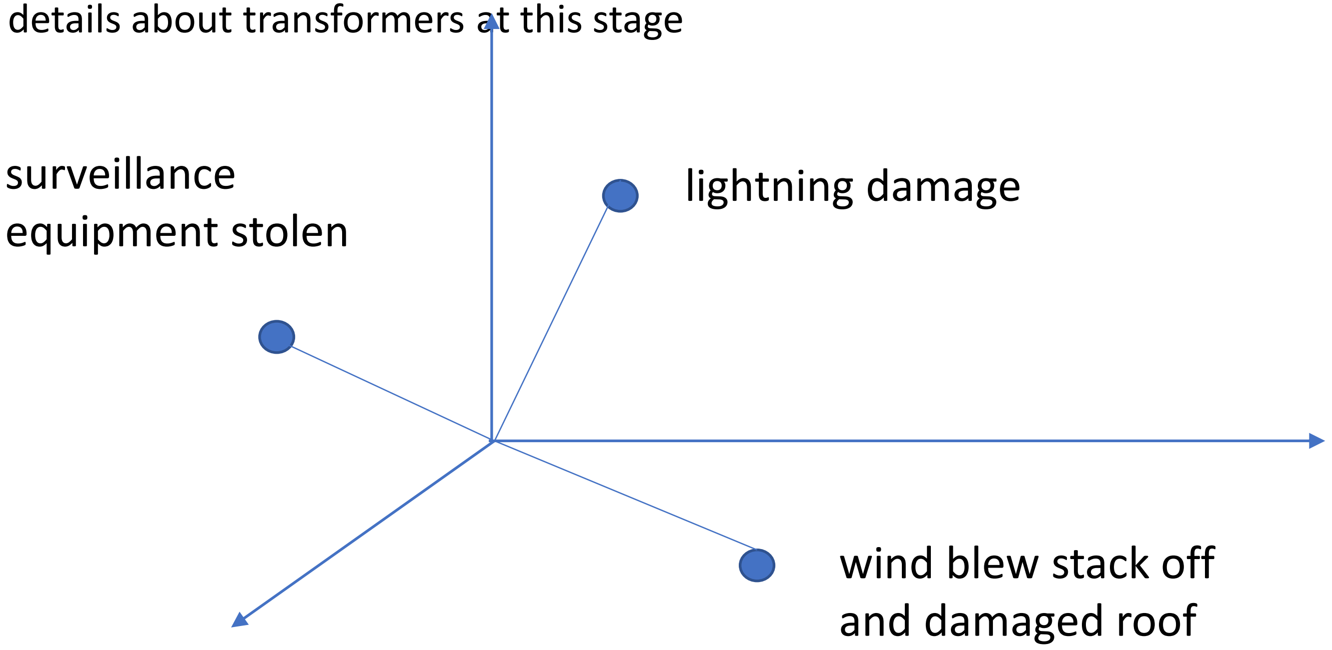
# High-level approach

How to fit a supervised model, when the feature space are words?

➔ Second idea: Embed the sentences in a low-dimensional space, such that there is some logic when vectors are close to each other

🤗 ➔ **Transformers** are models that do that embedding. And recently, it has been shown that those embedings are really good, compared to older models some years ago.

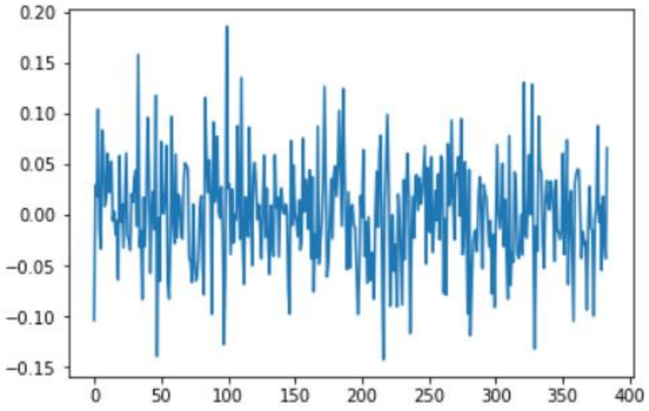
➔ We do not go into details about transformers at this stage



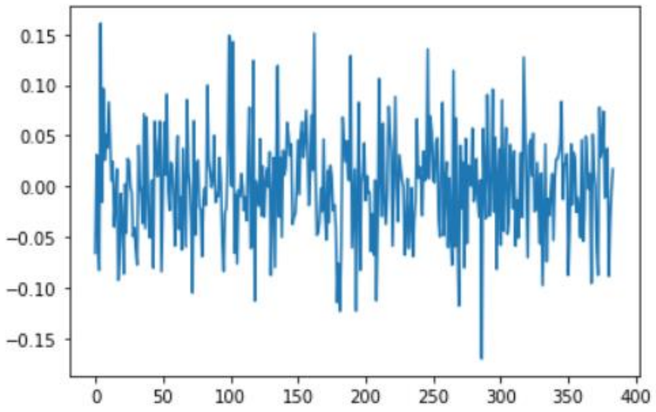
# Features

- x: 384 dimensional feature vector, all vectors of unit length
- Y: peril types (labels)

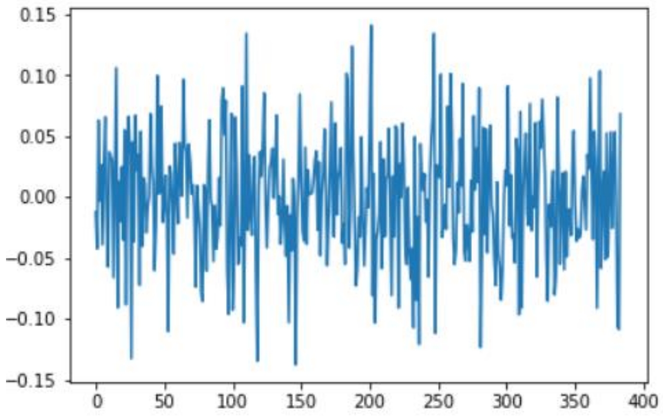
lightning damage




surveillance equipment stolen




wind blew stack off and damaged roof



# Results

## Dummy classifier

| actual class \ predicted class | Vandalism | Fire | Lightning | Wind | Hail | Vehicle | WaterNW | WaterW | Misc |
|---|---|---|---|---|---|---|---|---|---|
| Vandalism | 310 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Fire | 46 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Lightning | 123 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Wind | 107 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Hail | 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Vehicle | 227 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| WaterNW | 67 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| WaterW | 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Misc | 103 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## Logistic Regression classifier

| actual class \ predicted class | Vandalism | Fire | Lightning | Wind | Hail | Vehicle | WaterNW | WaterW | Misc |
|---|---|---|---|---|---|---|---|---|---|
| Vandalism | 296 | 1 | 0 | 1 | 0 | 6 | 0 | 0 | 6 |
| Fire | 3 | 32 | 3 | 0 | 0 | 2 | 0 | 1 | 5 |
| Lightning | 1 | 0 | 114 | 1 | 0 | 1 | 0 | 1 | 5 |
| Wind | 5 | 0 | 5 | 93 | 1 | 1 | 0 | 1 | 1 |
| Hail | 1 | 0 | 0 | 2 | 14 | 1 | 0 | 0 | 0 |
| Vehicle | 12 | 0 | 0 | 2 | 0 | 209 | 3 | 1 | 0 |
| WaterNW | 10 | 0 | 1 | 0 | 0 | 0 | 23 | 29 | 4 |
| WaterW | 1 | 0 | 0 | 2 | 0 | 1 | 5 | 29 | 0 |
| Misc | 20 | 1 | 4 | 1 | 0 | 12 | 1 | 2 | 62 |

# Python Code

## Using google Colab providing the infrastucture

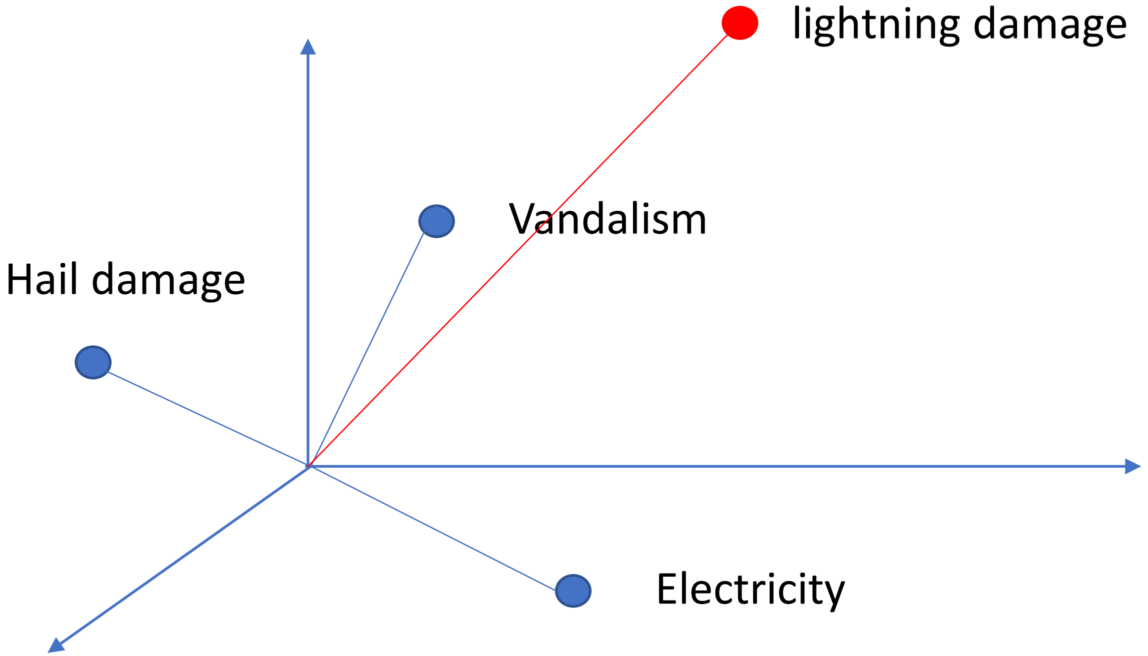
```
# load the model and the tokenizer
model_name = "distilbert-base-uncased«
tokenizer = AutoTokenizer.from_pretrained(model_name)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = AutoModel.from_pretrained(model_name).to(device)
# define a function to tokenize a batch
def tokenize(batch):
 return tokenizer(batch["Description"], truncation=True, padding=True, max_length=12)
# apply the function to the whole dataset
ds = ds.map(tokenize, batched=True)
ds = ds.map(extract_sequence_encoding, fn_kwargs={"model": model}, batched=True, batch_size=16)
x_train, y_train, x_test, y_test = get_xy(ds, "mean_hidden_state", "labels")
# fit a logarithmic regression classifier to the encoded texts
clf = logistic_regression_classifier(x_train, y_train, c=0.2)
```

Given the infrastructure, just a few lines of code are needed!

# Unsupervised classification using similarity

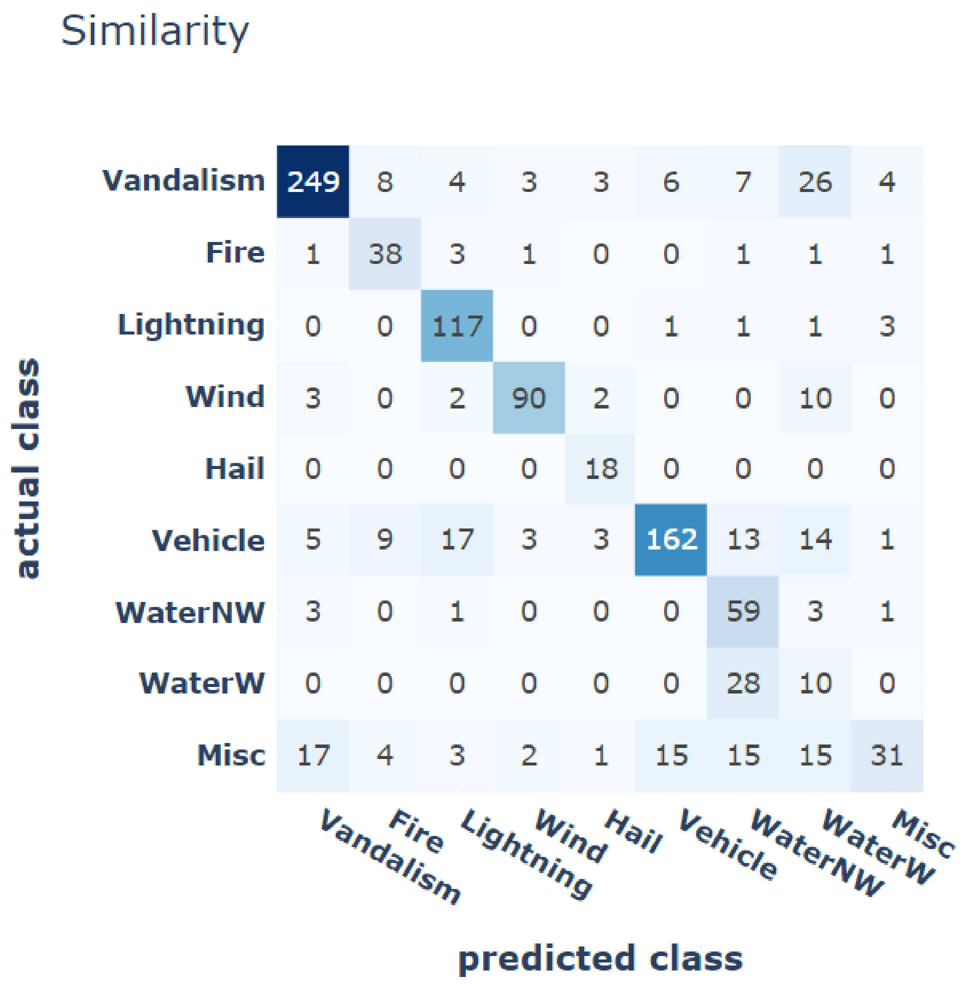
# High-level approach

- Every claims description is translated into a 384-dimensional vector with unit length
- Cosine similarity, which is the dot product of two embedding vectors, each normalized to unit length
- The peril type with the highest score is selected.



# Results

| Peril Type | Candidate expressions |
| --- | --- |
| Vandalism | Vandalism, Glass, Theft |
| Fire | Fire damage |
| Lightening | Lightning damage |
| Wind | Wind damage |
| Hail | Hail damage |
| Vehicle | Damage cause by a vehicle |
| WaterNW | Water damage |
| WaterW | Weather damage, Ice |
| Misc | Electricity, power surge |



# Conclusions

# Conclusions

- Transformers
  - Useful in situations of small data
  - Useful in situations with no labels
  - Transformer models are relatively new
  - Results are good due to progress in the language models used
  - Business problems which could not be solved 5 years ago are nowadays feasable
  - Few lines of codes
  - Computationally intensive. Platform with GPU support recommended.
- Tutorial available here, and corresponding Python notebooks here.
- www.actuarialdatascience.org

# Appendix

# Transformers

- Neural network architecture developed by Google researchers in 2017.
- Uses word embeddings and self-attention layers to understand words in their context.
- Quickly became dominant for achieving state-of-the art results on many NLP tasks.

- BERT (Bidirectional Encoder Representations from Transformers) is a Transformer encoder architecture, introduced in 2019
- Multilingual DistilBERT, derived from BERT: 134 million parameters, pre-trained on Wikipedia in 104 different languages
- Multilingual alternatives: XLM, XLM-RoBERTa, …

- Easy-to use Python library and model hub provided by 🤗 Huggingface (https://huggingface.co/)

# References


- Actuarial Applications of Natural Language Processing Using Transformers: Case Studies for Using Text Features in an Actuarial Context, A. Troxler, J. Schelldorfer, 2022, arXiv:2206.02014
- Statistical Foundations of Actuarial Learning and its Applications, M.V. Wüthrich and M. Merz, 2023, Springer Actuarial
- Frees, E.W. (2020). Loss data analytics. An open text authored by the Actuarial Community. https://openacttexts.github.io/
- Tunstall, L., von Werra, L., Wolf, T. (2022). Natural language processing with transformers. O'Reilly Media, Inc.

# Acknowledgements

**People:**

- All members of the SAA working group
- Dr. Andrea Ferrario
- Dr. Tobias Fissler
- Dr. Roger Hämmerli
- Mara Nägelin
- Dr. Alexander Noll
- Dr. Simon Renzmann
- Ron Richman
- Dr. Robert Salzmann

**Insititutions:**

- Swiss Association of Actuaries (SAA)
- RiskLab at ETH Zurich
- MobiLab for Analytics at ETH Zurich

**Companies:**

- Swiss Re