Classical Approaches vs. LLMs: A Comparative Study

STA 6908 – Independent Study

Spring 2025

Mina Akhondzadeh

SUMMARY

- Comparison of TF-IDF features and embeddings from fine-tuned LLMs
- Error analysis of highest misclassified categories using PCA, t-SNE, Monte Carlo Dropout, and LDA on the embeddings of the fine-tuned LLM
- Fine-tuned LLMs vs. zero-shot classification on AG-News and Amazon Reviews datasets
- Explored conditional clustering with LLMs using Hugging Face and OpenAI APIs

Final Thoughts:

Zero-shot classification is the slowest and delivers the lowest performance.
Logistic Regression on TF-IDF features is the fastest, with performance that is often comparable.
Fine-tuned LLMs achieve the best performance, though they require slightly more time.
For conditional clustering, **OpenAI's GPT-4** performs best, as it allows you to define the condition directly in the prompt. Beyond that, the best results come from LLMs specifically fine-tuned for the given condition.

CHAPTER 1: AG News - LLMs vs Classical Models

About data:

- Sampled 10,000 articles from over 1 million news articles
- Widely used as a text classification benchmark
- Labels: World, Sports, Business, and Sci/Tech
- There is an already fine-tuned BERT-based model trained on this dataset
- Balanced categories

Models:

1. Multiclass Logistic Regression on TF-IDF
2. Zero-shot classification using an unsupervised LLM with predefined target labels
3. A pre-fine-tuned BERT model (`textattack/bert-base-uncased-ag-news`)
4. Fine-tuned a `bert-base-uncased` model on our dataset

1. Multiclass Logistic Regression on TF-IDF:

- Preprocessing text:

raw_text → lowercase → remove special characters → tokenize → remove stopwords → lemmatize → TF-IDF → ML model

- Lemmatization: Reducing words to their base form
- For lemmatizing using `WordNetLemmatizer` from `nltk`
- Train a Multiclass Logistic Regression model (`max_iter=500, penalty="l2"(default), multi_class="multinomial"`)

Method Characteristic:

- Computationally efficient
- Fast to run

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

2. Zero-shot Classification(`facebook/bart-large-mnli`):

- Unsupervised
- with predefined target labels
- `facebook/bart-large`
    - Trained using Denoising autoencoding objectives (token masking, token deletion, sentence shuffling)
    - This means BART learns to reconstruct original text from a corrupted version
    - **pre-trained** on general corpora like **books** and **Wikipedia**

- `facebook/bart-large-mnli`
    - **fine-tuned** on the **MNLI** dataset for the natural language inference task

2. Zero-shot Classification(`facebook/bart-large-mnli`):

What is Natural Language Inference (NLI)?

NLI is the task where the model looks at two sentences (a *premise* and a *hypothesis*) and decides if the hypothesis is:

- **Entailed** by the premise (definitely true),
- **Contradicted** (definitely false), or
- **Neutral** (could be true or false, we can't tell).

What is MNLI?

MNLI stands for *Multi-Genre Natural Language Inference*.
It's a benchmark dataset used to train models on **entailment reasoning**

Entailment Reasoning example

Premise	Label	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction	The man is sleeping.
An older and younger man smiling.	neutral	Two men are smiling and laughing at the cats playing on the floor.
A soccer game with multiple males playing.	entailment	Some men are playing a sport.

Source:https://nlpprogress.com/english/natural_language_inference.html

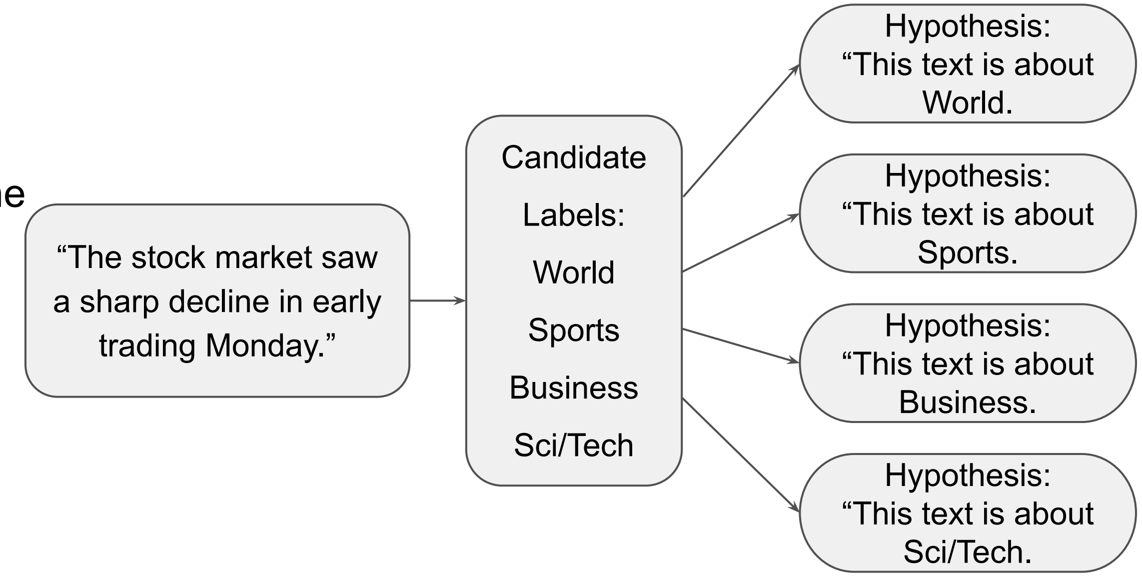
Teaching models to understand relationships between two sentences.

2. Zero-shot Classification(`facebook/bart-large-mnli`):

Entailment Reasoning Example in Classification Context

How likely the label (”hypothesis”) is true, given the input text?

The model checks which label best fits the meaning of the text. (The label with the highest entailment score)



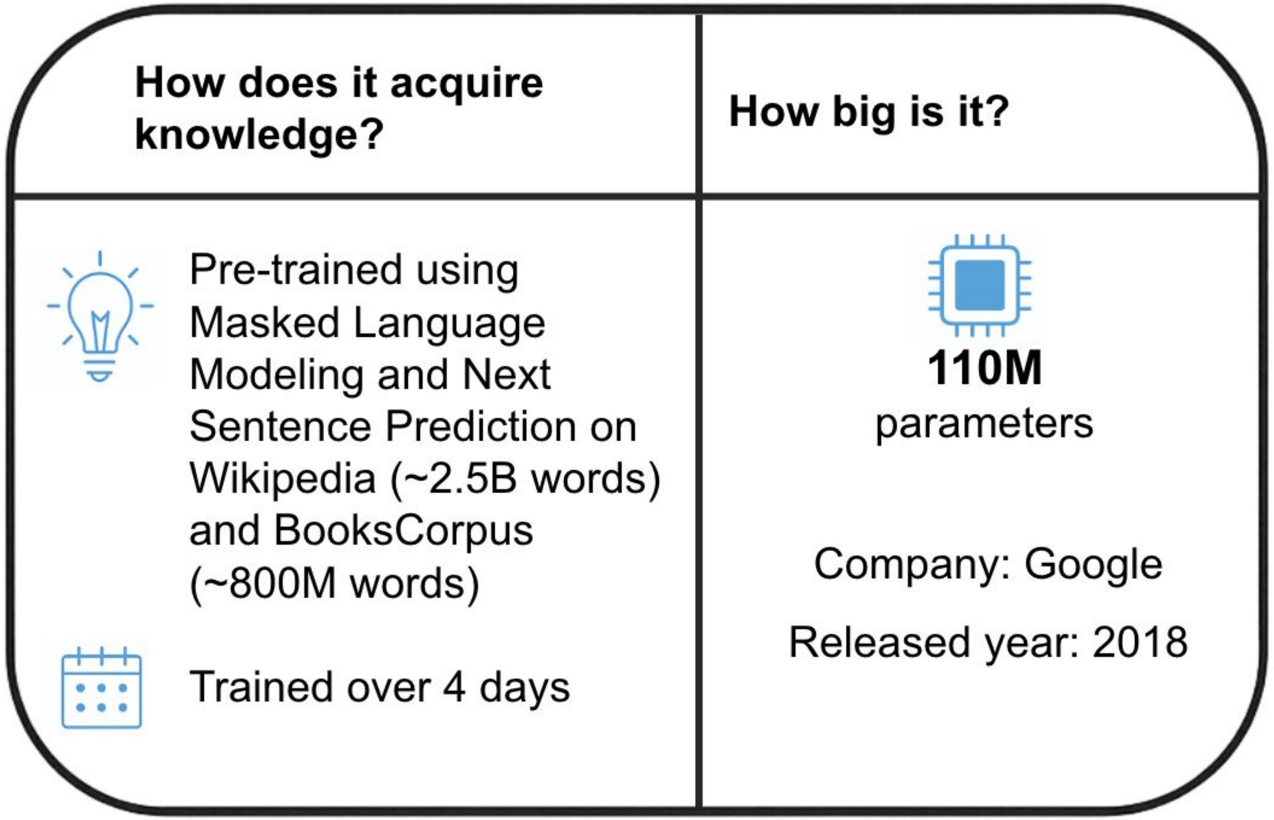
How does this help with classification?

The trick behind **zero-shot classification** is to **reframe a classification problem** as an **entailment problem**.

3. Pre-fine-tuned BERT model on AG News Dataset (`textattack/bert-base-uncased-ag-news`):

- Load the model and tokenizer using Hugging Face’s transformers library.
- Perform batch-wise inference on the test set.
  - Tokenize
  - Feed model
  - Output logits → predicted class labels

BERT Model



3. Pre-fine-tuned BERT model on AG News Dataset (`textattack/bert-base-uncased-ag-news`):

BERT Model

How does it acquire knowledge?	How big is it?
Pre-trained using Masked Language Modeling and Next Sentence Prediction on Wikipedia (~2.5B words) and BooksCorpus (~800M words) Trained over 4 days	**110M** parameters Company: Google Released year: 2018

What is Masked Language Modeling (MLM)?

Example:

"The stock market [MASK] sharply on Monday."
→ The model learns to predict "dropped".

What is Next Sentence Prediction (NSP)?

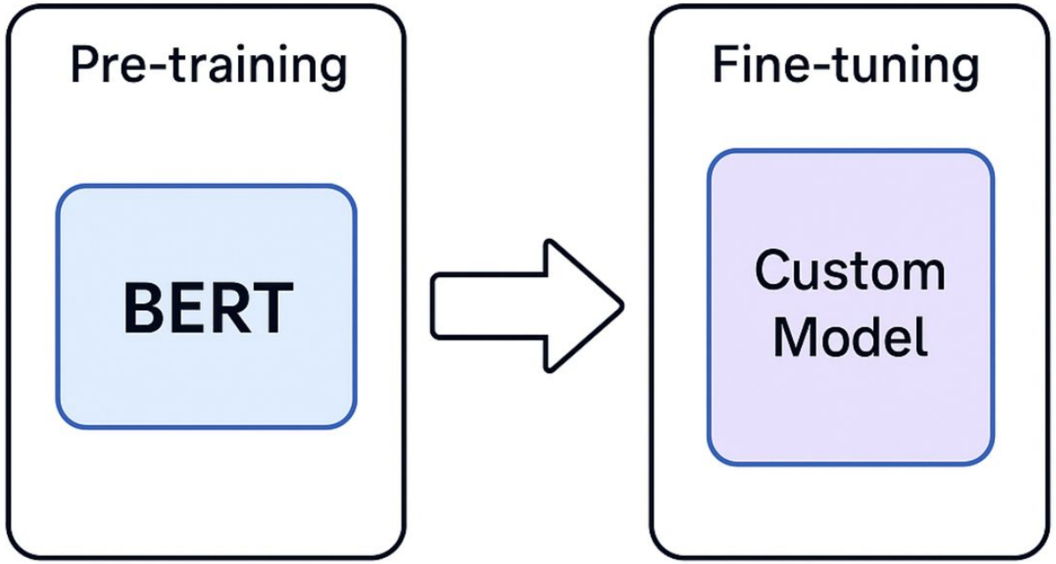
Example:

Sentence A: "She opened the door."
Sentence B: "She walked into the room." → Likely follows
Sentence B: "The sun is hot." → Probably unrelated

4. Fine-tune BERT model on AG News Dataset (`textattack/bert-base-uncased`):

One of the great things about LLMs?

The pre-training process is separated from the fine-tuning process.



We can customize them for our specific use cases

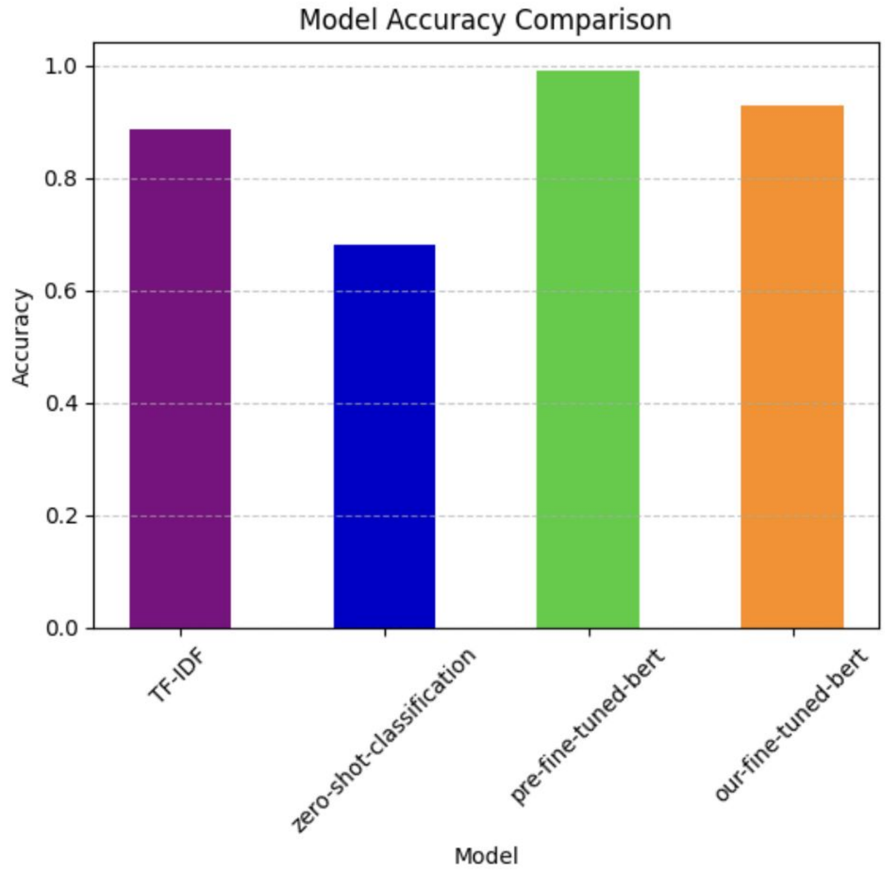
4. Fine-tune BERT model on AG News Dataset (`textattack/bert-base-uncased`):

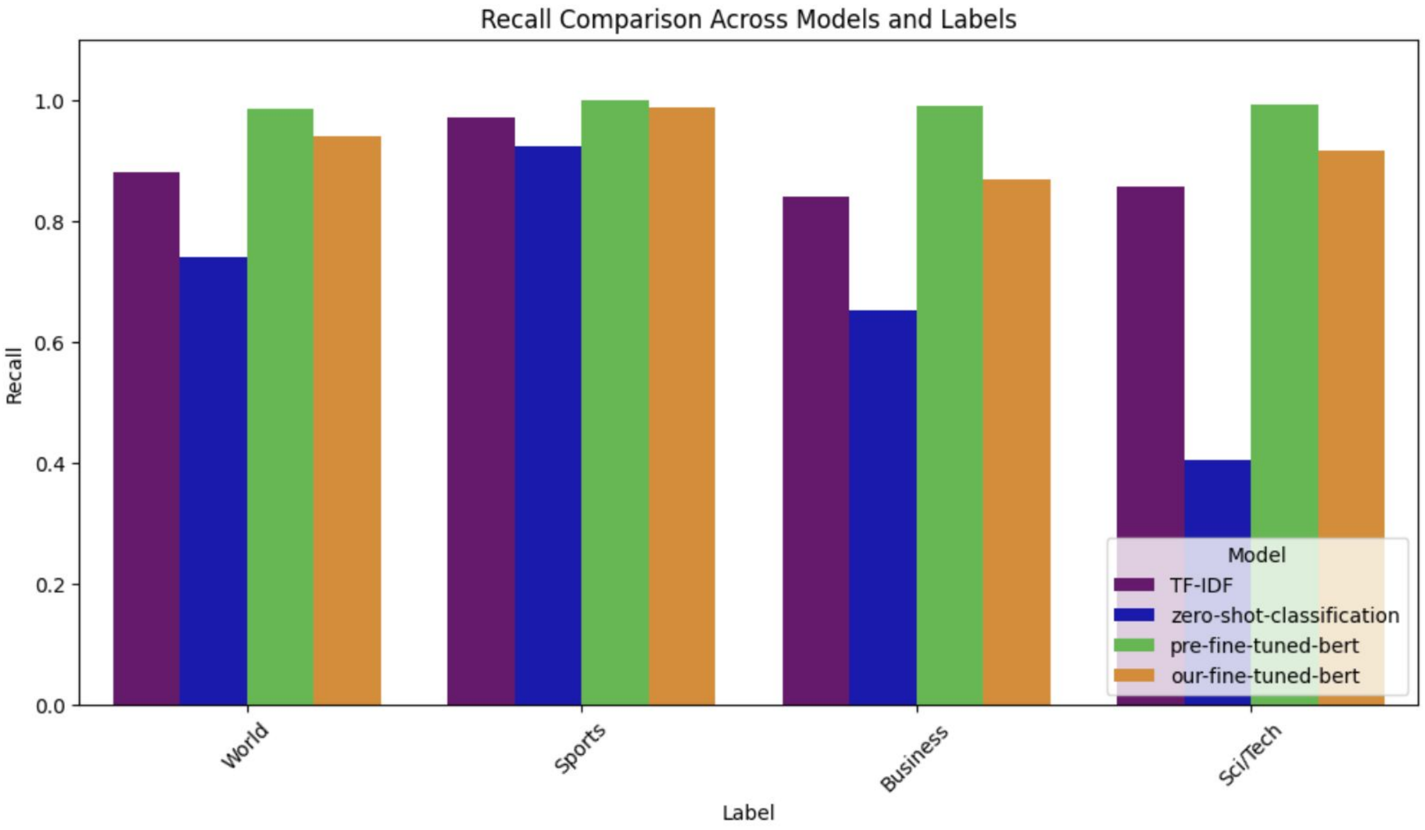
PROCESS:

- Prepared AG News data using the **Hugging Face datasets library**
- **Tokenized text** using the BERT base uncased tokenizer with truncation and padding
- **Loaded BERT model** for sequence classification with **4 output labels**
- Configured training using **Hugging Face's Trainer API**
  set learning rate(3e-5), batch size(16), number of epochs(5), weight decay(0.1), and warm-up steps
- **Fine-tuned** the model on tokenized training data
- Predictions on the test set
- Output logits → predicted class labels (using argmax)

Evaluation:

Balanced accuracy: the average of sensitivity + specificity
Plots are for the test set





CHAPTER 2: Amazon Reviews Classification + Error Analysis

About Data:

I used the **2023 Amazon Reviews** dataset from McAuley Lab, selecting 5,000 samples from each of the following categories (originally 10–15M reviews each) to maximize variation in **topic**, **tone**, and **sentiment**:

- Beauty and Personal Care
- Books
- Electronics
- Grocery and Gourmet Food
- Toys and Games
- Office Products

Models:

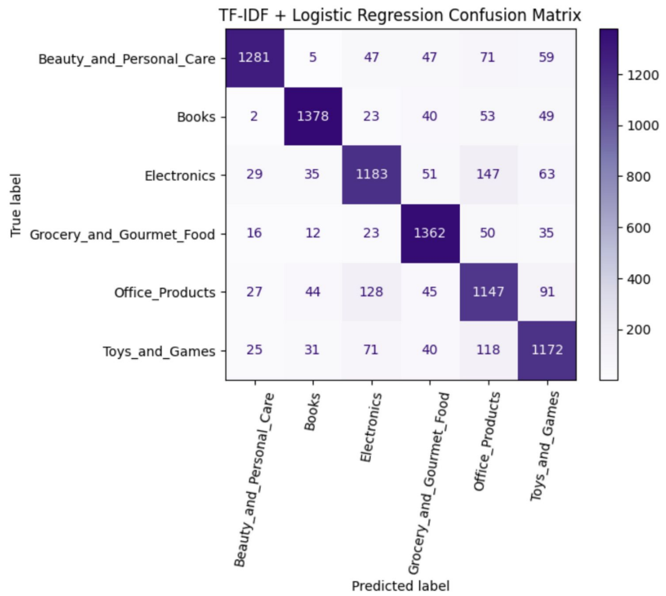
Classical Model:

- Multiclass **Logistic Regression** on **TF-IDF** feature space
- **Random Forest** on **TF-IDF** feature space

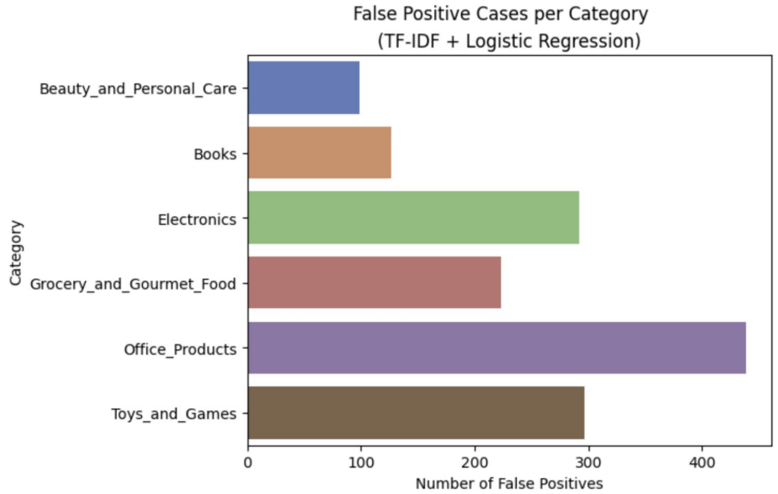
LLM:

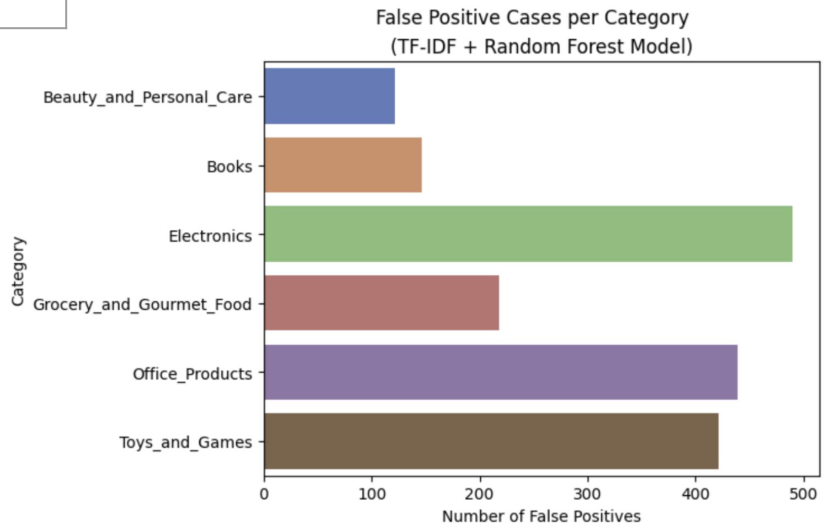
- Fine-tuned **`distilbert/distilbert-base-uncased`** model on the dataset

Evaluation: Logistic Regression performed better at classifying the Amazon review categories.



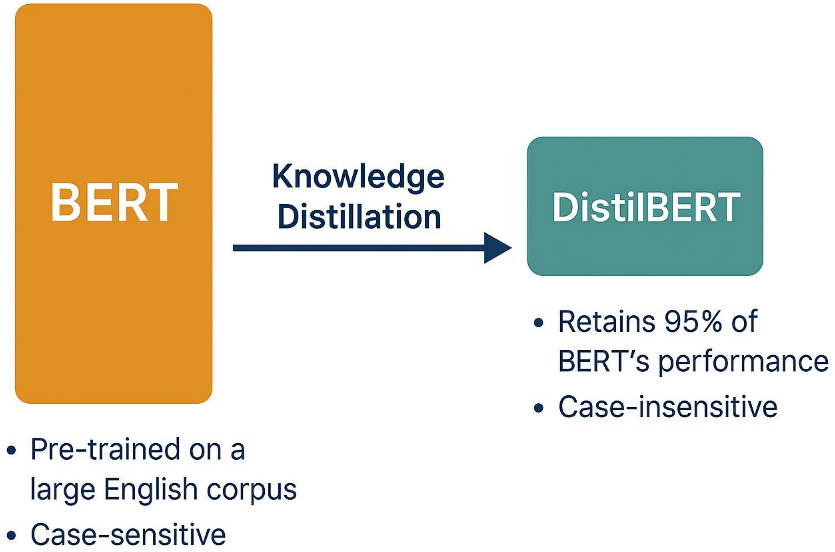
MODEL	Logistic Regression	Random Forest
TRAIN ACCURACY	0.87	0.86
TEST ACCURACY	0.84	0.80





3. Fine-tuned `distilbert/distilbert-base-uncased` on Amazon Reviews

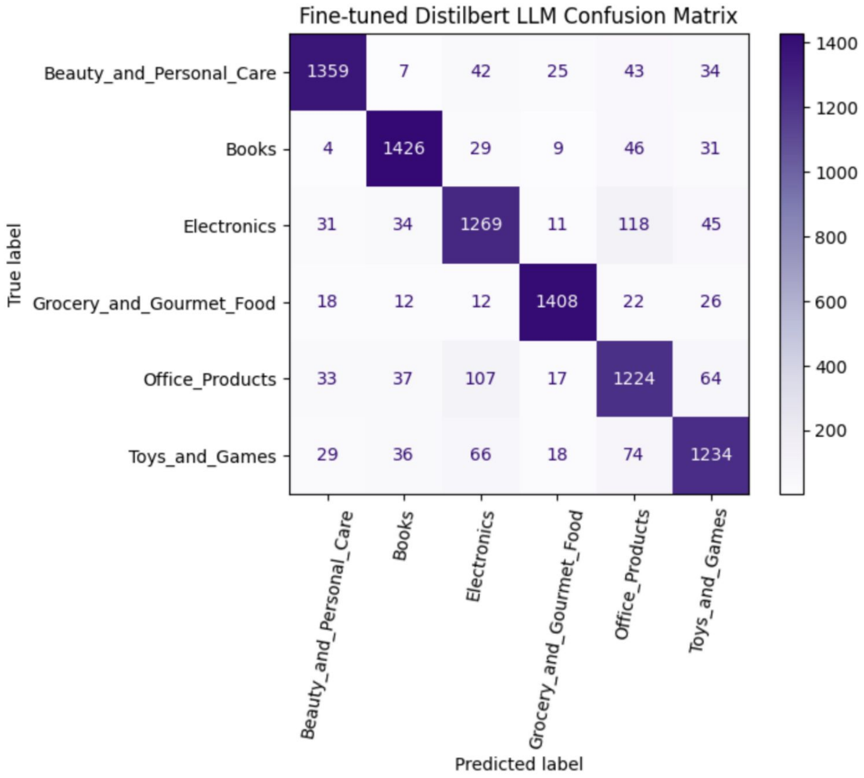
- Fine-tuned **DistilBERT** model on Amazon Reviews
- DistilBERT is a smaller, faster version of BERT
- Trained using **knowledge distillation** (BERT = teacher, DistilBERT = student)
- Student mimics the teacher's **soft predictions**, not just hard labels
- Goal: maintain performance while reducing size and increasing speed
- DistilBERT retains ~95% of BERT's accuracy

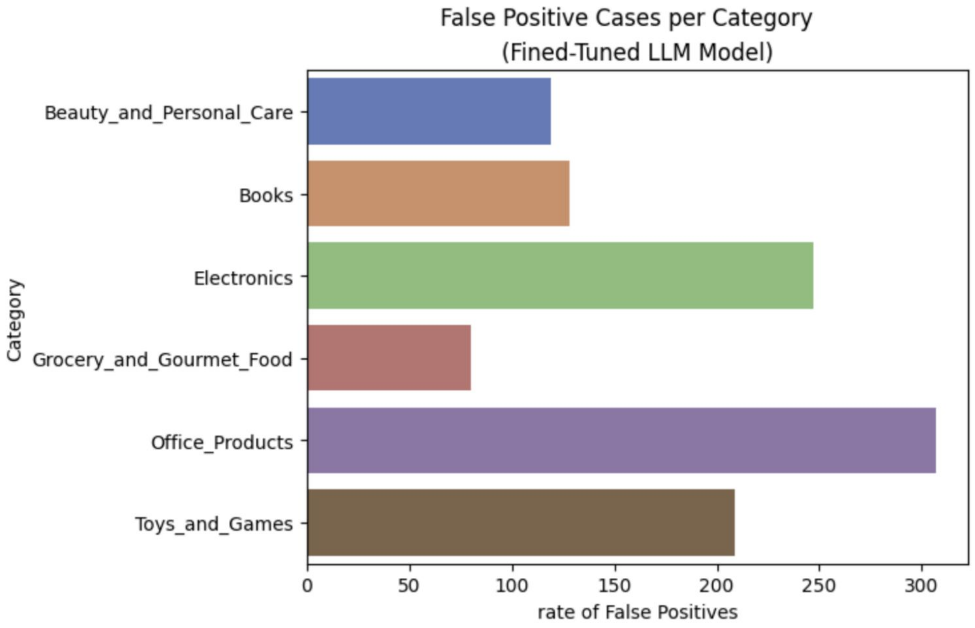


3. Fine-tuned `distilbert/distilbert-base-uncased` on Amazon Reviews

- Overall **better performance**
- Some signs of **overfitting**
- Continued difficulty distinguishing between **Office_Products** and **Electronics** categories

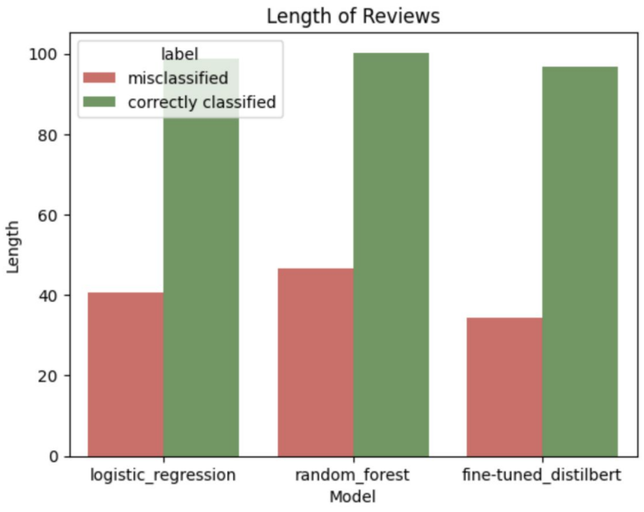
MODEL	Logistic Regression	Random Forest	fine-tuned LLM
TRAIN ACCURACY	0.87	0.86	0.95
TEST ACCURACY	0.84	0.80	0.88





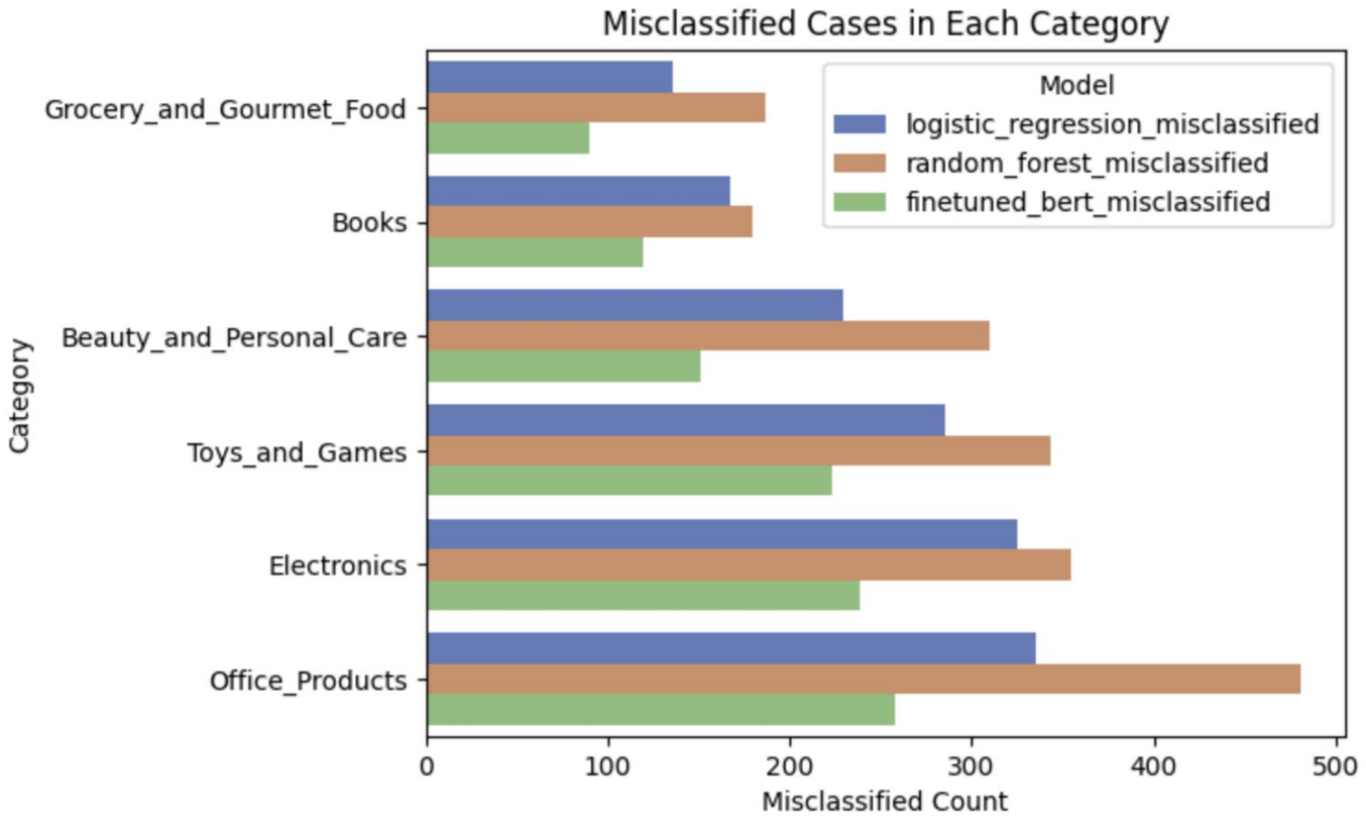
4. Error Analysis

- Shorter reviews are misclassified more often.



- Reviews that are misclassified by all three models tend to be too broad and general.

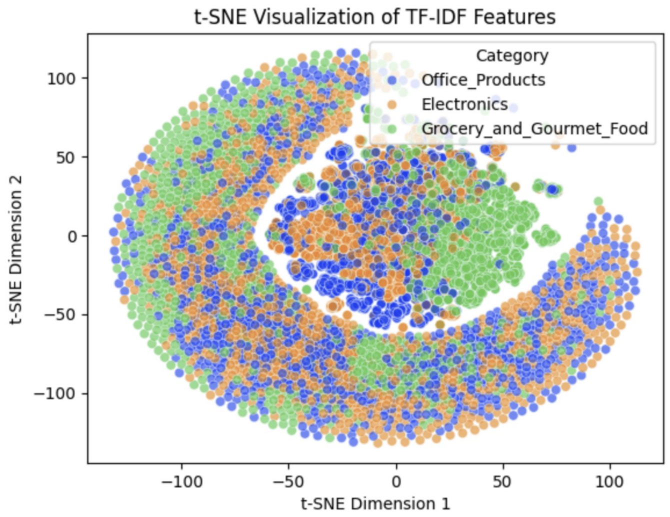
- Finding a pattern for `Office_Products` and `Electronics` categories is the hardest.



5. Narrowing Down to Three Categories:

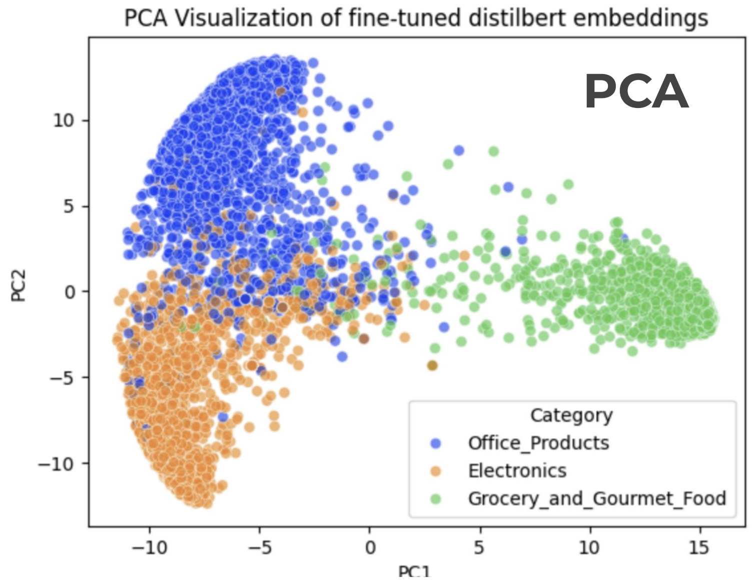
- `Electronics`
- `Office_Products`
- `Grocery_and_Gourmet_Food`

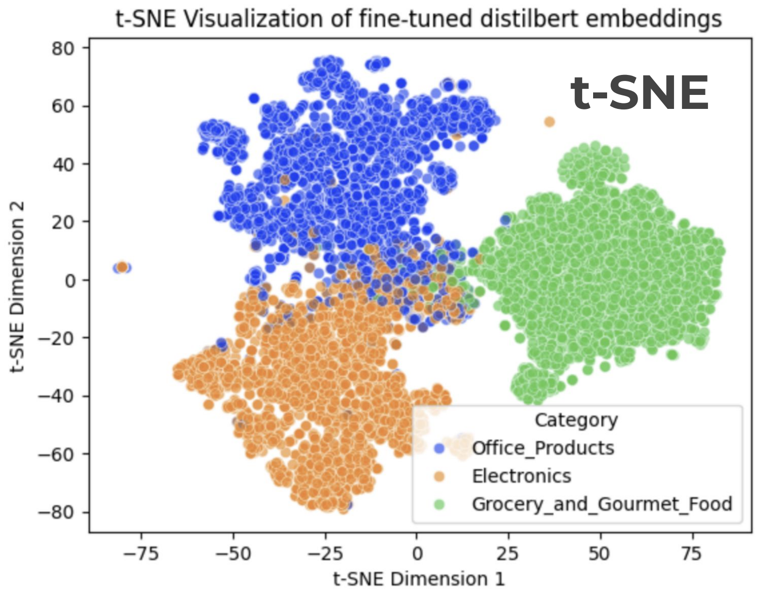
TF-IDF



PCA Doesn't Work Well on TF-IDF but t-SNE would.

LLM Embeddings





❖ The t-SNE plot of **TF-IDF features** shows some green clustering, but **orange and blue are highly mixed**.

❖ Both PCA and t-SNE of **LLM embeddings** show **clearer separation for green**, with some **overlap between blue and orange.**

6. Estimating Uncertainty in Fine-Tuned DistilBERT Using Monte Carlo Dropout

randomly turn off neurons during evaluation → creates a distribution of outputs → estimate prediction uncertainty

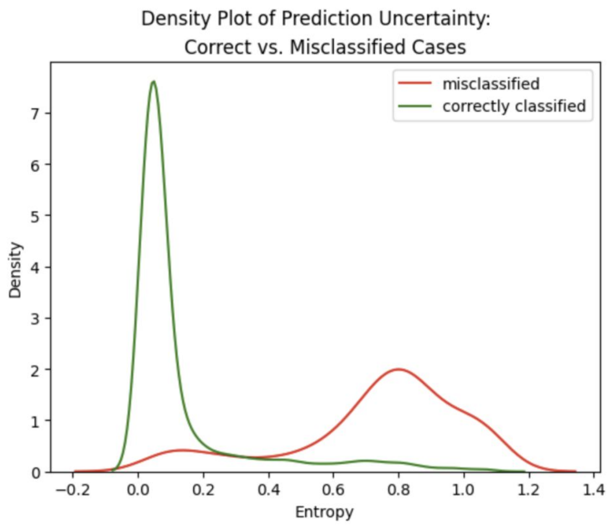
- ➔ Modified the model to **keep dropout active during evaluation** (`module.train()`)
- ➔ Performed **20 forward passes**
- ➔ Collected the **logits → probabilities**
- ➔ **Averaged probabilities** over the 20 iterations for each sample
- ➔ Calculated **entropies**

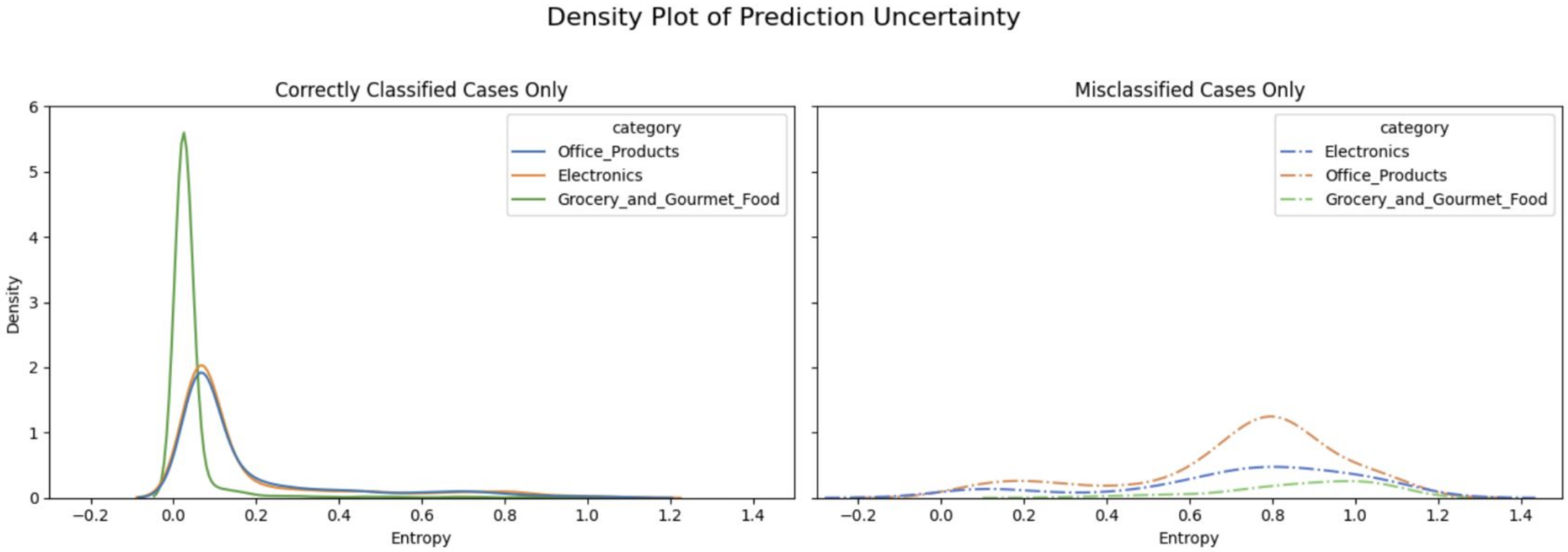
$$\text{entropy}_j = -\sum_i p_{ji} \log(p_{ji} + 10^{-10})$$

- j: each sample (rows in `probs_mean`).
- i: indexes the classes (columns in `probs_mean`).
- p_{ji}: the predicted probability of class i for sample j.

6. Estimating Uncertainty in Fine-Tuned DistilBERT Using Monte Carlo Dropout

Overall **uncertainty distribution** for correct vs. misclassified predictions, measured using **entropy**

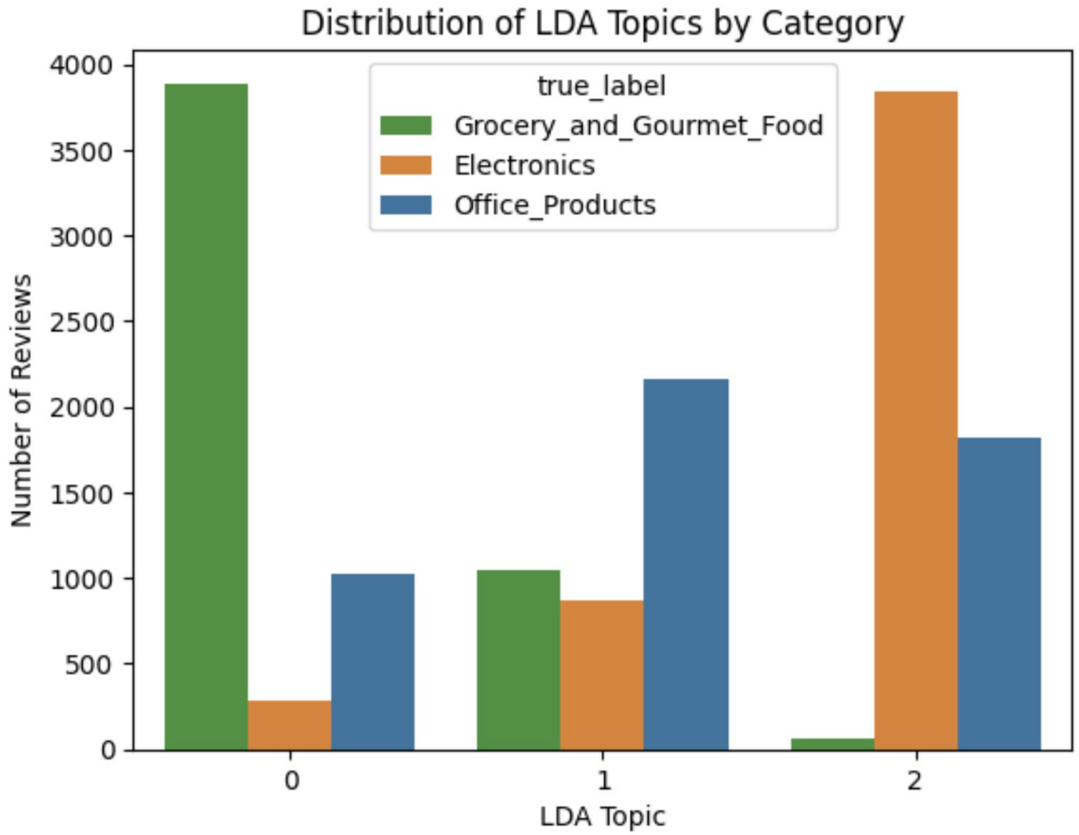




7. Topic Modeling Latent Dirichlet Allocation (LDA)

Each **topic** is a probability distribution over **words**
Each **document** is a mixture of **topics**

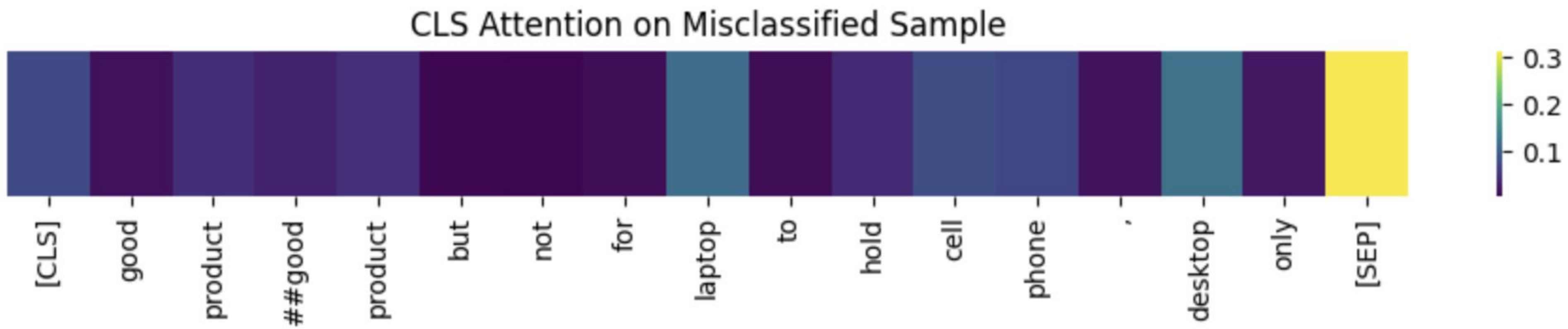
`Office_Products` and `Electronics` are struggling.



8. DistilBERT [CLS] Attention Visualization: Correct vs. Misclassified Case

- As a final step, I explored something called **attention weights** to better understand how the model makes decisions.
- In transformer models like DistilBERT, attention tells us **which words the model focuses on the most** when trying to make a prediction.
- This heatmap shows how much attention the special `[CLS]` token gives to each word in the sentence.
- The `[CLS]` token is what the model uses to summarize the whole input and make the final prediction.

Attention helps us peek into the model’s reasoning

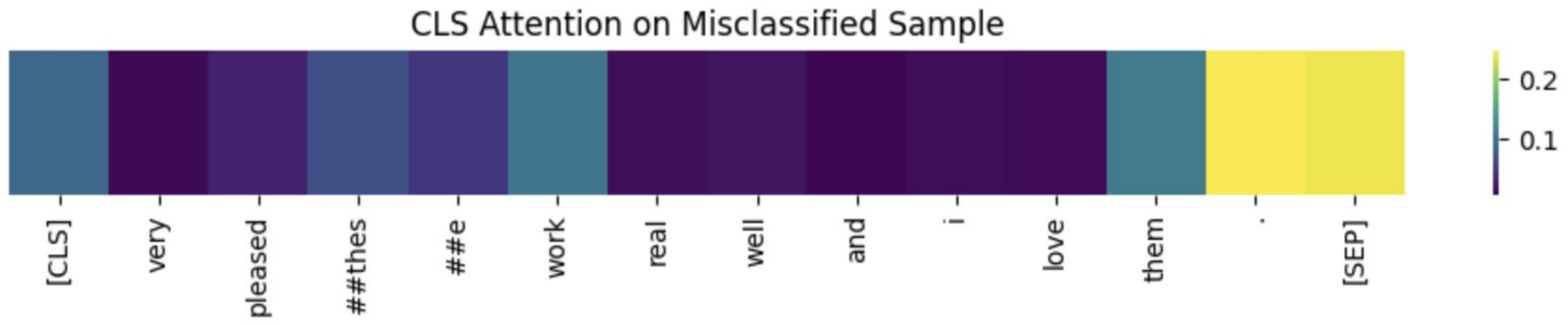


`Good product Good product but not for laptop to hold cell phone, desktop only`

`true label:` **`Office_Products`**

`misclassified as:` **`Electronics`**

8. DistilBERT [CLS] Attention Visualization: Correct vs. Misclassified Case

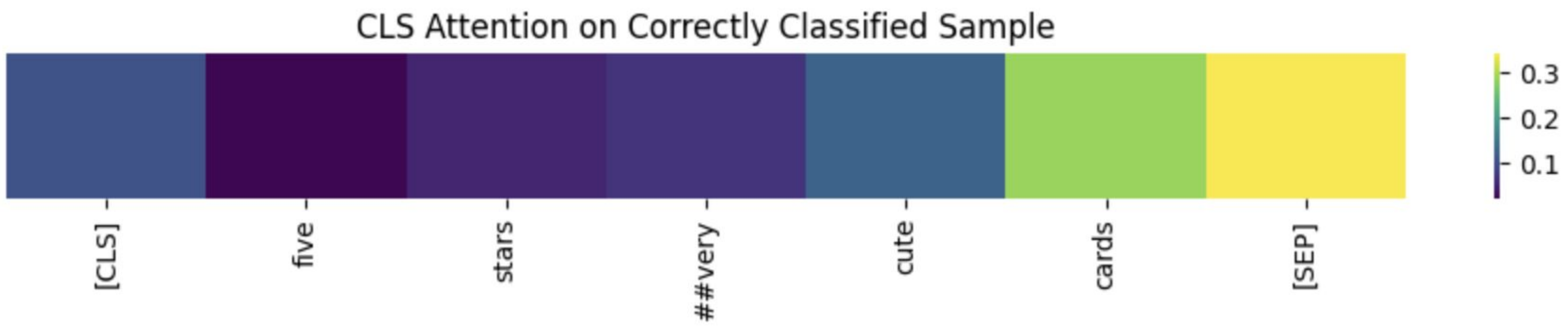


Very pleased These work real well and I love them.

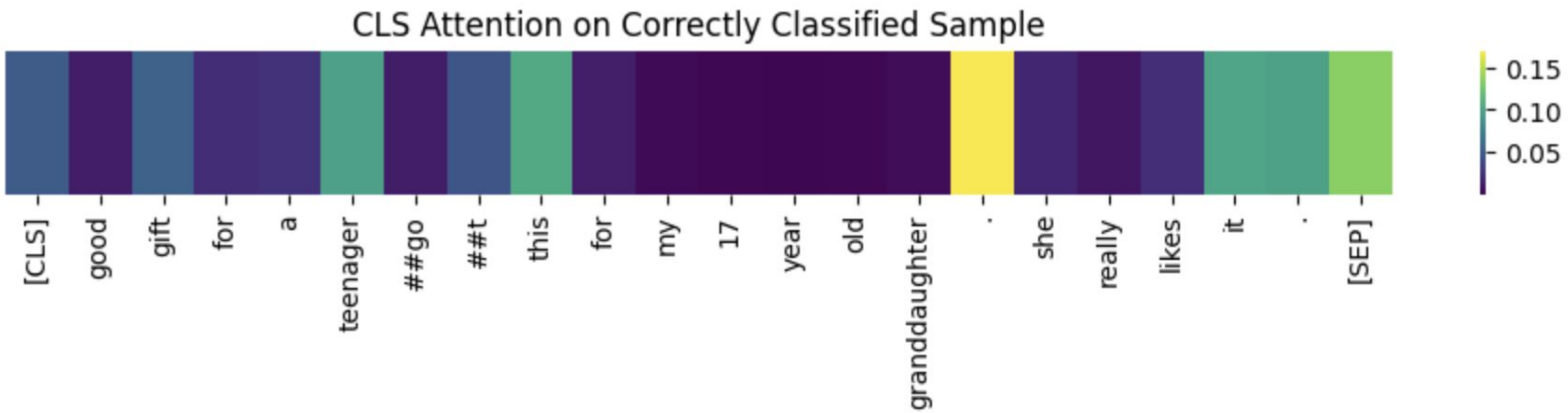
true label: **Electronics**

misclassified as: **Office_Products**

8. DistilBERT [CLS] Attention Visualization: Correct vs. Misclassified Case



Five Stars VERY cute cards
true label: **Office_Products**
classified as: **Office_Products**

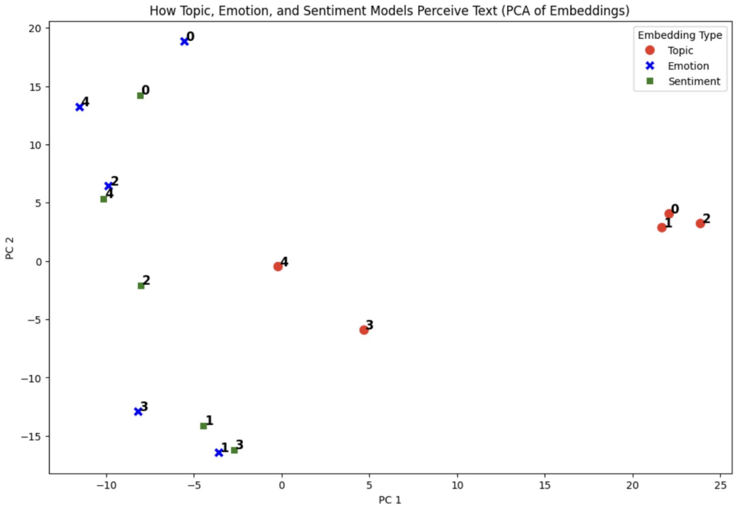


Good gift for a teenager Got this for my 17 year old granddaughter. She really likes it.
true label: **Electronics**
classified as: **Electronics**

CHAPTER 3: Conditional Clustering on Text

About Data:

To better understand how models make clustering decisions, I provided five texts that could be clustered based on topic, emotion, and sentiment.



Models:

OpenAI Conditional Clustering Using Prompts:

- Model **`gpt-4`**
- Embeddings of the model **`text-embedding-ada-002`**

Task-specific LLMs:

- `cardiffnlp/tweet-topic-21-multi`
  Fine-tuned on a **tweet topic classification** task with 19 categories (e.g., politics, sports, etc.)

- `j-hartmann/emotion-english-distilroberta-base`
  Fine-tuned on **emotion classification**, using datasets like GoEmotions (joy, anger, fear, etc.)

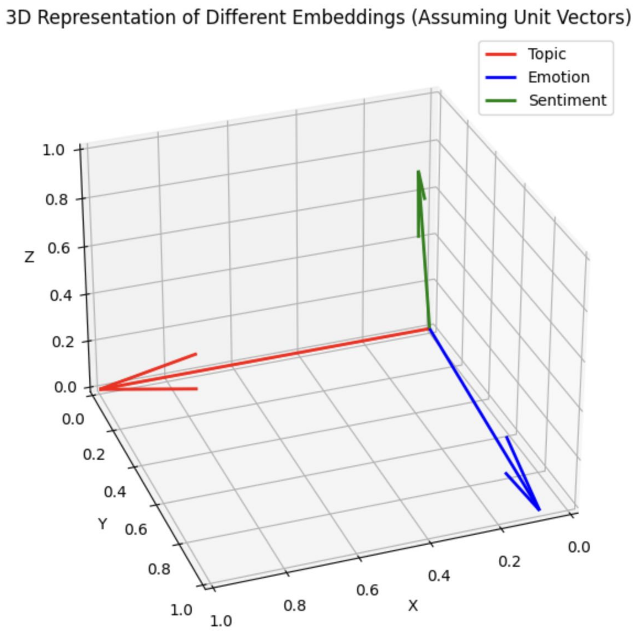
- `cardiffnlp/twitter-roberta-base-sentiment`
  pretrained on tweets then fine-tuned on **sentiment analysis** tasks (positive, neutral, negative)

Classical Model:

- **K-Means** Clustering on **TF-IDF** features

3. Task-specific LLMs

- **for topic classification:**
  `cardiffnlp/tweet-topic-21-multi` model
- **for emotions classification:**
  `j-hartmann/emotion-english-distilroberta-base` model
- **Task-specific LLMs for sentiment classification:**
  `cardiffnlp/twitter-roberta-base-sentiment` model



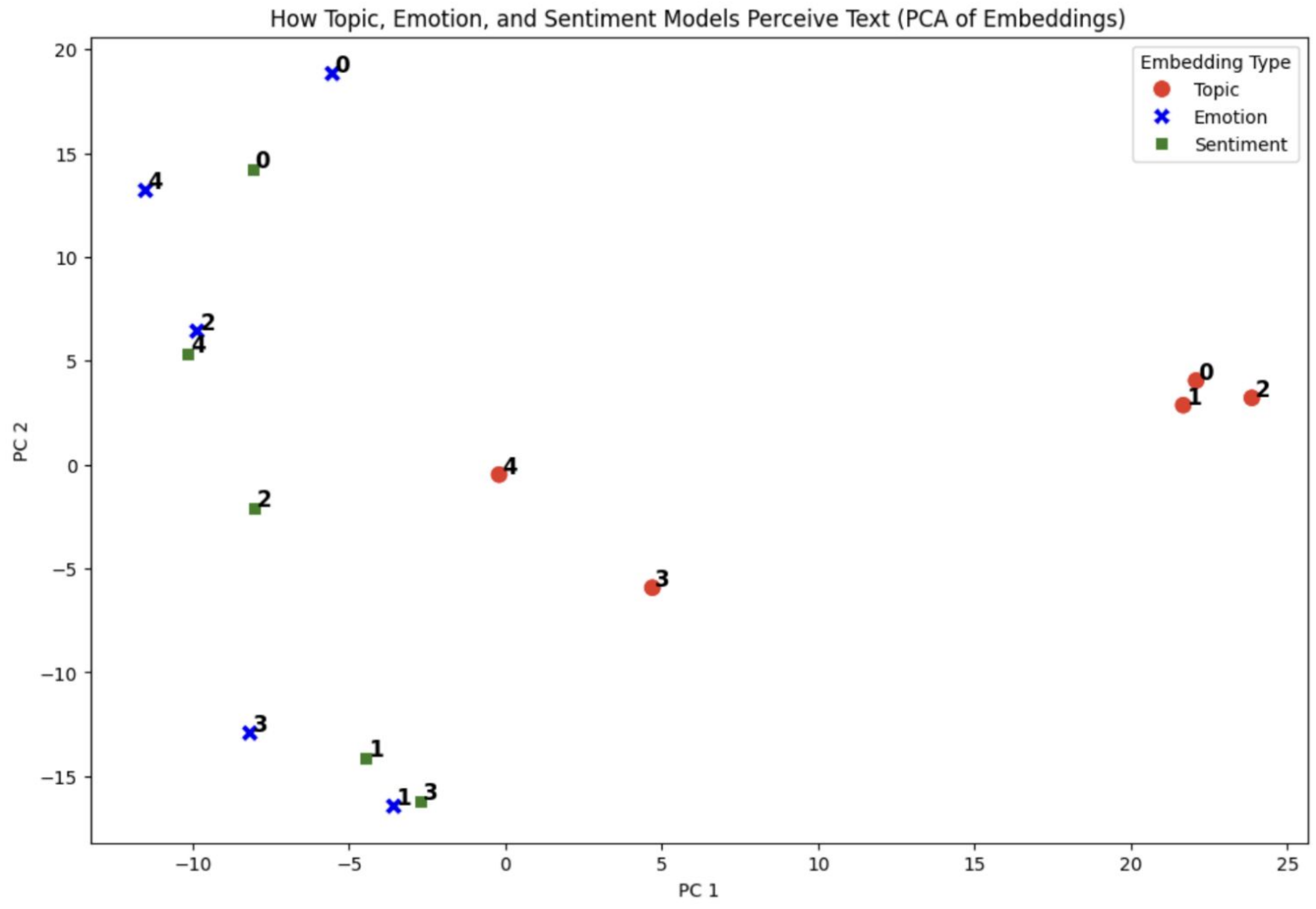
Embeddings Comparison

Each model has its own vector space and encoding system; therefore, the similarities between their embeddings shouldn't be very high.
I used `cosine_similarity` from `sklearn.metrics.pairwise` to calculate the similarity between each pair of criteria.

- Similarity (Topic vs. Emotion): 0.08
- Similarity (Topic vs. Sentiment): 0.17
- Similarity (Emotion vs. Sentiment): 0.33

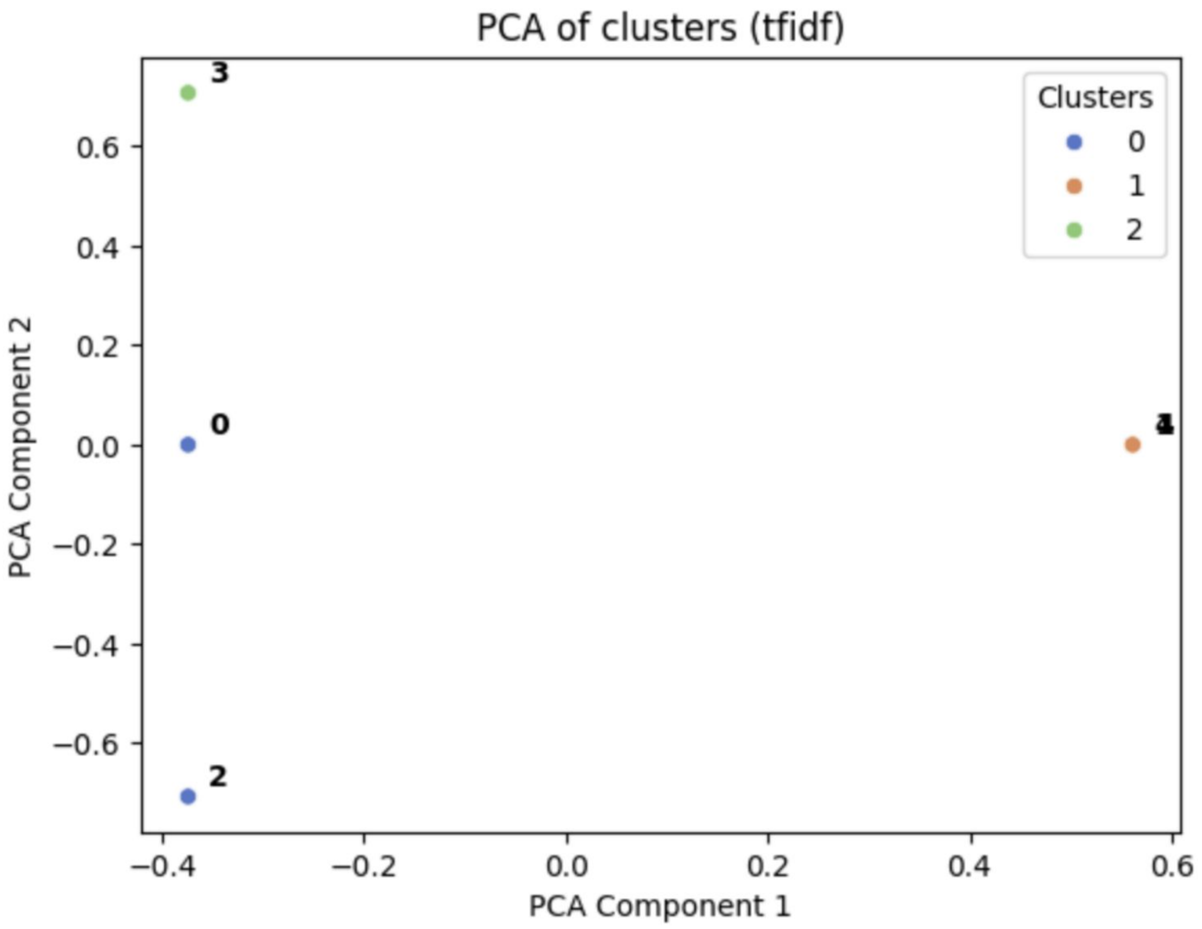
$$\text{cosine\_similarity}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

How Topic, Emotion, and Sentiment Models Perceive Text: A PCA Visualization of Their Embeddings



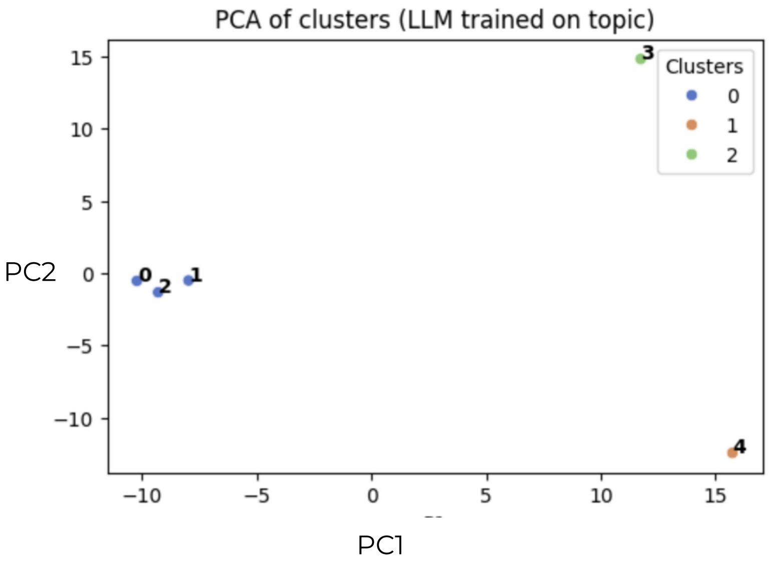
0. "The soccer team celebrated their championship victory with a parade."
1. "The marathon runner collapsed just meters away from the finish line."
2. "Basketball players need both speed and strategy to dominate the court."
3. "The stock market experienced a sharp decline today, causing panic among investors."
4. "NASA just launched a new telescope to explore distant galaxies."

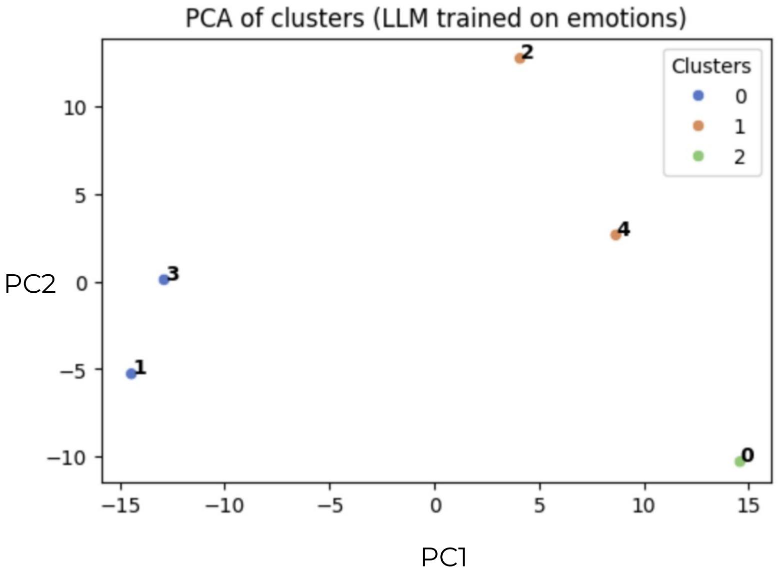
4. K-Means Clustering on TF-IDF Feature Space

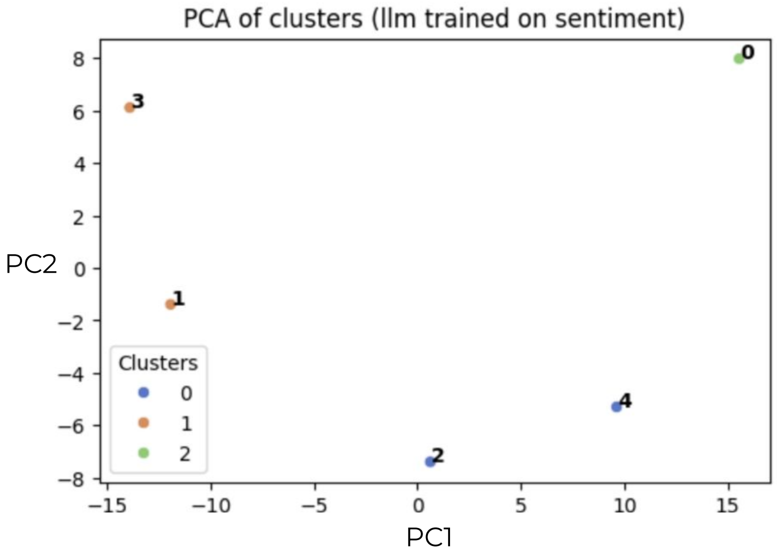


- To establish a traditional baseline
- Number of clusters: **3**
- Datapoints 0, 2, and 3 share the same PC1 value but differ in PC2
  (**sport, sport, finance**)
  (**positive, neutral, negative**)
- Datapoints 1 and 4 surprisingly have identical values for both PC1 and PC2
  (**sport, science**)(**negative, neutral**)
- TF-IDF is based purely on word frequency

5. K-Means Clustering on LLM Embeddings







What we expected to see for **topic** clustering:

- 0, 1, 2 (sport)
- 3 (finance)
- 4 (science)

What we expected to see for **emotions** clustering:

- 0 (happy)
- 1, 3 (sad)
- 2, 4 (neutral)

What we expected to see for **sentiment** clustering:

- 0 (positive)
- 1, 3 (negative)
- 2, 4 (neutral)

THANK YOU!

model	accuracy	set
TFIDF + Linear Regression	0.864889	test
TFIDF + Linear Regression	0.912571	train
LLM	0.903556	test
LLM	0.964762	train
LDA	0.805111	test
LDA	0.817619	train

References:

AG News Dataset

Zero-Shot Learning in Modern NLP

What is BERT? An Intro to BERT Models on DataCamp

DistilBERT base model

What is an attention mechanism?

Hugging Face Model Outputs

Hugging Face BERT