Lab 7 - Classification - DL - Emotion Analysis

🧭 Overview

Emotion analysis is a natural language processing (NLP) method for detecting human emotions in text.
We use **GoEmotions** (https://research.google/blog/goemotions-a-dataset-for-fine-grained-emotion-classification/), a dataset developed by Google containing over 58,000 Reddit comments labeled with 27 emotions, and a **pretrained model** from Hugging Face: `SamLowe/roberta-base-go_emotions`. The objective is to detect emotions like *joy*, *fear*, *anger*, or *optimism* in text and visualize emotion distributions.

🧰 Learning objectives

By the end of this notebook, you will be able to:

- Understand what emotion analysis is and how it differs from sentiment analysis
- Apply a pretrained transformer model to classify emotions in text
- Analyze emotions in finance-related examples
- Visualize emotion distributions with Python

🧩 1. Introduction: Emotions and Financial Text

Emotion analysis helps uncover **psychological signals** in market-related language.
While sentiment analysis typically measures polarity (*positive/negative/neutral*), emotion analysis provides a **finer-grained understanding** — identifying *fear*, *joy*, *surprise*, *anger*, and more.

Applications in finance:

- Detecting *fear* or *optimism* in market news
- Understanding emotional tone in *analyst reports* or *investor comments*
- Supporting *risk management* and *trading strategies*

Examples:

- "Investors panic as inflation hits record highs." ⟶ *Fear*
- "Analysts celebrate Tesla's unexpected growth." ⟶ *Joy, Surprise*
- "Traders express frustration over delayed regulations." ⟶ *Anger*

🤖 2. The GoEmotions Dataset and Model

GoEmotions is one of the largest emotion-labeled datasets for English text.
The model used in this seminar — `SamLowe/roberta-base-go_emotions` — is based on the **RoBERTa** architecture, fine-tuned to classify 28 categories (27 emotions + neutral).

🧪 3. Hands-On: Emotion Classification with RoBERTa-GoEmotions

Step 1. Setup

```
# Install the required libraries
%pip install transformers torch pandas matplotlib
```

Step 2. Import the libraries

```
from transformers import pipeline
import pandas as pd
import matplotlib.pyplot as plt
```

Step 3. Load the pretrained model

```
emotion_model = pipeline("text-classification",
                         model="SamLowe/roberta-base-go_emotions",
                         return_all_scores=True)
```

Step 4. Create a small dataset of financial texts

```
data = {
    "text": [
        "Investors are thrilled by Apple's record-breaking quarter.",
        "Uncertainty rises as oil prices continue to fluctuate.",
        "The market reacted with anger after unexpected tax changes.",
        "Optimism grows following the central bank's positive outlook.",
        "Analysts express disappointment over lower-than-expected growth."
    ]
}

df = pd.DataFrame(data)
df
```

	text
0	Investors are thrilled by Apple's record-break...
1	Uncertainty rises as oil prices continue to fl...
2	The market reacted with anger after unexpected...
3	Optimism grows following the central bank's po...
4	Analysts express disappointment over lower-tha...

Step 5. Run emotion analysis

```
results = emotion_model(df["text"].tolist())
results
```

```
 {'label': 'caring', 'score': 0.003384896088391542},
 {'label': 'confusion', 'score': 0.0016964878886938095},
 {'label': 'curiosity', 'score': 0.00445155706256628},
 {'label': 'desire', 'score': 0.004518173169344664},
 {'label': 'disappointment', 'score': 0.0016120824730023742},
 {'label': 'disapproval', 'score': 0.0022149917203933},
 {'label': 'disgust', 'score': 0.0009651023428887129},
 {'label': 'embarrassment', 'score': 0.0008445940911769867},
 {'label': 'excitement', 'score': 0.5693874955177307},
 {'label': 'fear', 'score': 0.001600584713742137},
 {'label': 'gratitude', 'score': 0.008788608014583588},
 {'label': 'grief', 'score': 0.0007271160720847547},
 {'label': 'joy', 'score': 0.3262365162372589},
 {'label': 'love', 'score': 0.00985987950116396},
 {'label': 'nervousness', 'score': 0.0015814534854143858},
 {'label': 'optimism', 'score': 0.0038949092850089073},
 {'label': 'pride', 'score': 0.009040424600243568},
 {'label': 'realization', 'score': 0.0026892346795648336},
 {'label': 'relief', 'score': 0.004433504305779934},
 {'label': 'remorse', 'score': 0.0003256065247114748},
 {'label': 'sadness', 'score': 0.0013003380736336112},
 {'label': 'surprise', 'score': 0.008467145264148712},
 {'label': 'neutral', 'score': 0.05156151950359345}],
[{'label': 'admiration', 'score': 0.0037147735711187124},
 {'label': 'amusement', 'score': 0.0014660198939964175},
 {'label': 'anger', 'score': 0.0018405100563541055},
 {'label': 'annoyance', 'score': 0.008919747546315193},
 {'label': 'approval', 'score': 0.04707979038357735},
 {'label': 'caring', 'score': 0.011174790561199188},
 {'label': 'confusion', 'score': 0.8294706344604492},
 {'label': 'curiosity', 'score': 0.023966435343027115},
 {'label': 'desire', 'score': 0.002885820111259818},
 {'label': 'disappointment', 'score': 0.00897995661944151},
 {'label': 'disapproval', 'score': 0.0181267149746418},
 {'label': 'disgust', 'score': 0.0015860950807109475},
 {'label': 'embarrassment', 'score': 0.0017266470240429044},
 {'label': 'excitement', 'score': 0.003521013306453824},
 {'label': 'fear', 'score': 0.006334247533231974},
 {'label': 'gratitude', 'score': 0.0028967217076569796},
 {'label': 'grief', 'score': 0.0007839421159587 8},
 {'label': 'joy', 'score': 0.005298870615661144},
 {'label': 'love', 'score': 0.005821148864924908},
 {'label': 'nervousness', 'score': 0.010258561000227928},
 {'label': 'optimism', 'score': 0.020984698086977005},
 {'label': 'pride', 'score': 0.0005865990533493459},
 {'label': 'realization', 'score': 0.028637241572141647},
 {'label': 'relief', 'score': 0.0017651956295594573},
 {'label': 'remorse', 'score': 0.0026305078063160 18},
 {'label': 'sadness', 'score': 0.0032212608493864536},
 {'label': 'surprise', 'score': 0.0025772633962333202},
 {'label': 'neutral', 'score': 0.2161082923412323}],
```

```
 {'label': 'caring', 'score': 0.0026419858913868666},
```

```
 {'label': 'confusion', 'score': 0.003478448837995529},
```

```
emotion_scores = [{r['label']: r['score'] for r in result} for result in results]
df = pd.concat([df, pd.DataFrame(emotion_scores)], axis=1)
df.head()
```

	text	admiration	amusement	anger	annoyance	approval	caring	confusion	curiosity	desire	...	love
0	Investors are thrilled by Apple's record-break...	0.160311	0.004949	0.002569	0.005086	0.026368	0.003385	0.001696	0.004452	0.004518	...	0.009860
1	Uncertainty rises as oil prices continue to fl...	0.003715	0.001466	0.001841	0.008920	0.047080	0.011175	0.829471	0.023966	0.002886	...	0.005821
2	The market reacted with anger after unexpected...	0.001116	0.002037	0.354785	0.292093	0.010090	0.002642	0.003478	0.008243	0.001779	...	0.001847
3	Optimism grows following the central bank's po...	0.041883	0.001220	0.000445	0.001939	0.344639	0.058192	0.001852	0.001280	0.011055	...	0.003359
4	Analysts express disappointment over lower-tha...	0.004128	0.001347	0.006442	0.081443	0.009895	0.003588	0.008469	0.005366	0.007025	...	0.006622

5 rows × 29 columns

Double-click (or enter) to edit

```
df.to_excel("emotions.xlsx", index=False)
```

```
def extract_emotions(text):
    results = emotion_model(text)
    return {r['label']: r['score'] for r in results[0]}

df["emotions"] = df["text"].apply(extract_emotions)
df.head()
```

	text	admiration	amusement	anger	annoyance	approval	caring	confusion	curiosity	desire	...	nervousr
0	Investors are thrilled by Apple's record-break...	0.160311	0.004949	0.002569	0.005086	0.026368	0.003385	0.001696	0.004452	0.004518	...	0.001
1	Uncertainty rises as oil prices continue to fl...	0.003715	0.001466	0.001841	0.008920	0.047080	0.011175	0.829471	0.023966	0.002886	...	0.010
2	The market reacted with anger after unexpected...	0.001116	0.002037	0.354785	0.292093	0.010090	0.002642	0.003478	0.008243	0.001779	...	0.005
3	Optimism grows following the central bank's po...	0.041883	0.001220	0.000445	0.001939	0.344639	0.058192	0.001852	0.001280	0.011055	...	0.001
4	Analysts express disappointment over lower-tha...	0.004128	0.001347	0.006442	0.081443	0.009895	0.003588	0.008469	0.005366	0.007025	...	0.007

5 rows × 30 columns

Step 6. Aggregate and visualize emotion scores

```
emotions_df = pd.DataFrame(df["emotions"].to_list()).fillna(0)
```

```
avg_emotions = emotions_df.mean().sort_values(ascending=False).head(10)
```

```
# Plot the most frequent emotions
avg_emotions.plot(kind="bar", figsize=(10, 4), title="Top Emotions in Financial Texts")
plt.xlabel("Emotions")
plt.ylabel("Average Probability")
plt.show()
```