**EXERCISE 1 P. 88**

**Topic modeling** was performed on the *newsgroups* corpus using two complementary methods: **LDA (Latent Dirichlet Allocation)** and **NMF (Non-negative Matrix Factorization)**. According to the lecture (pages 70–81), both approaches help uncover the latent semantic structure of a large collection of documents by grouping similar terms together under broader topics.

**Data Pre-processing**

Before modeling, the entire corpus was:

* converted to lowercase,
* stripped of punctuation,
* and vectorized using **CountVectorizer** for LDA and **TF-IDF Vectorizer** for NMF.

This standard text cleaning pipeline aligns with the course recommendations for preparing text data for unsupervised learning.

**LDA (Latent Dirichlet Allocation)**

Following the lecture example, LDA was applied on the document-term matrix built with word counts. LDA models each document as a mixture of various topics, with each topic defined by a probability distribution over words. The model was set to find **4 topics**, whose top words were extracted and manually interpreted to assign human-readable topic names:

* **Topic #1: Daily Life** covering casual discussion terms like :like, just, don, think, time, car.
* **Topic #2: Communication** words like: edu, com, thanks, new, use suggesting online communication and technical exchanges.
* **Topic #3: Computers** technical words such as disk, drive, card, controller, scsi, related to hardware and storage.
* **Topic #4: Science & Space** highlighting terms such as space, nasa, science, data, pointing to scientific and space-related discussions.

The Wordclouds below show the 30 most significant terms per topic, where larger words have higher probability weights. This visual approach supports the course’s principle that visual topic interpretation improves understanding.

Une image contenant texte, Police, capture d’écran, Graphique

Le contenu généré par l’IA peut être incorrect.

**NMF (Non-negative Matrix Factorization)**

NMF was also applied to the same corpus but using a TF-IDF representation as recommended in the course. This method factorizes the term-document matrix into two parts:

* topics as combinations of words,
* and documents as combinations of topics.

NMF was set to extract 4 topics as well. After reviewing the top words, the topics were interpreted and labeled as follows:

* **Topic #1: Daily Discussion** general words like: don, like, think, people, time.
* **Topic #2: Apple Hardware** hardware and computer terms including: drive, disk, scsi, mac, card.
* **Topic #3: Debate & Skepticism** intellectual and debate-related words such as skepticism, chastity, shameful, edu.
* **Topic #4: Sports** words like game, team, year, hockey, players pointing to sports topics.

Again, Wordclouds were created to visually represent the relative weight of each term in its topic. This follows the course suggestion that visual tools help validate the coherence of discovered topics.

Une image contenant texte, Police, Graphique, graphisme

Le contenu généré par l’IA peut être incorrect.

**EXERCISE 2 P.99**

**Sentiment analysis** was performed on a tweets dataset using the **VADER** algorithm, as discussed in the lecture.

Before applying VADER, all tweets were cleaned to remove URLs, mentions, special characters, and punctuation. The text was also converted to lowercase to standardize the input. The cleaned tokens were joined back into full sentences so that the sentiment could be computed at sentence level, which matches the principle shown in the course for applying VADER to social media text.

A custom function was defined to analyze each tweet’s sentiment using VADER’s **compound score**, which represents the overall polarity of the text:

* If the compound score is >= 0.05, the tweet is classified as **Positive**.
* If the compound score is <= -0.05, the tweet is classified as **Negative**.
* Otherwise, the tweet is classified as **Neutral**.

The function then adds two new columns to the dataset:

* **sentiment\_label**: Positive, Neutral, or Negative.
* **sentiment\_score**: the compound polarity score computed by VADER.

This method provides a straightforward and explainable way to quantify the sentiment of social media content, which is suitable for short informal texts like tweets.

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.

**EXERCISE 3 P. 101**

In addition to lexicon-based sentiment analysis (VADER), a machine learning approach was applied using a pre-trained **Transformers** model from the transformers library.

A sample of 500 tweets was used. Each tweet was cleaned to remove URLs, mentions, numbers and punctuation, and converted to lowercase. To ensure compatibility with the BERT tokenizer’s input size limit (512 tokens maximum), all cleaned tweets were split into tokens and truncated if needed (250 tokens max), then joined back into full sentences using spaces as separators. This step ensures that the model receives proper sentences as input, in line with best practices discussed in the course.

The pipeline('sentiment-analysis') was used to classify each tweet as either POSITIVE or NEGATIVE. The model automatically returns a **confidence score** for each prediction. These results were saved as two new columns:

* **sentiment\_label**: the predicted polarity (POSITIVE or NEGATIVE).
* **sentiment\_score**: the model’s confidence for the predicted label.

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.

**EXERCISE 4 P.125**

In this task, a Large Language Model (Qwen-2.5-0.5B) was used to classify emails as either *SPAM* or *HAM*. The input dataset (*spam.csv*) contains raw email texts that needed to be automatically labeled. Following the instructions in the course session on Large Language Models (LLMs) for question answering and classification, the model was loaded directly from Hugging Face using the transformers library.

Each email was processed with a clear prompt engineering strategy: a structured *system-user* message format was created to make the LLM return only the class label. The system role defined the assistant’s behavior (*“You are Qwen, a helpful assistant…”*) while the user role contained the explicit instruction *“Classify this email as SPAM or HAM. Only answer with SPAM or HAM.”* and the full email text. This ensured the model’s output stayed focused and avoided extra explanations.

The predicted class for each email was extracted, cleaned and saved in a new column named *llm\_label*. This approach demonstrates how modern LLMs can be guided by precise instructions to perform text classification on real-world data, complementing traditional supervised methods.

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.