Assignment 3. NLP

(Upload to Moodle and defend by 25.10.2025)

If the assignment is uploaded and defended late,
–20% is deducted from each task,
i.e., if all are completed, the **maximum score is 80 points**.

Task 1. (25 points) Development and testing of sentiment analysis

Problem statement
Input: a sentence (or review) for evaluating the emotional tone.
Output: sentiment of the sentence on the scale: *negative, neutral, positive.*

Implement two approaches:

1. **Lexicon-based** — based on a dictionary of sentiment words (for example, *SentiWords*).
2. **Machine learning** — a classifier trained on a labeled dataset (for example, *IMDb*, *Amazon reviews*).

Work procedure
Step 1. Loading and preparing data

- Choose a dataset of reviews (*IMDb*, *Amazon product reviews*, or similar).
- Split the data into training, validation, and test sets.

Step 2. Text preprocessing

- Tokenization (splitting into words).
- Converting to lowercase.
- Removing stopwords.
- Lemmatization (normalization of words).

Step 3. Converting texts into vectors

- *Bag-of-Words* or *TF-IDF*.
- For the lexicon-based method — calculate a “sentiment score” according to the dictionary.

Step 4. Model training

- Train a classifier (*Logistic Regression* and *Naive Bayes*).
- For the lexicon-based approach, select thresholds for classification into 3 classes.

Step 5. Evaluation of quality

- Build a *confusion matrix.*

- Calculate the metrics: *accuracy*, *F1-score.*

Step 6. Error analysis

- Find examples of incorrectly classified sentences.
- Analyze the causes of errors: sarcasm, polysemous words, preprocessing errors.
- Propose improvements (for example, expanding the dictionary, using contextual models, handling negations).

Step 7. Visualization

- Build a *word cloud* for positive and negative reviews.
- Display the top important words (according to model weights).

Task 2. (25 points) Development and testing of an automatic summarization method

Problem statement
Input: 3 text files (attached to the assignment).
Output:

1. **Extractive summarization:**
   - Select 3 key sentences from each document.
   - Form a single combined list of 6 sentences for all documents.
   - Write a brief conclusion based on the results.
2. **Abstractive summarization (advanced version):**
   - Based on 5 documents, create a short abstract that should be:
     - shorter than the original documents,
     - grammatically coherent and correct,
     - containing the most important information.

Work procedure
Step 1. Data loading

- Use the provided 3 (or 5) text documents.
- Read them into the program (for example, using *open()* or *pandas*).

Step 2. Text preprocessing

- Sentence tokenization.
- Text cleaning (removal of special symbols, converting to lowercase).
- Optional: stopword removal, lemmatization.

Step 3. Extractive summarization

- Calculate the “importance” of sentences (for example, by *TF-IDF* or *TextRank*).
- Extract 3 sentences per document.
- Form the final list of 6 sentences (by maximum weight among all documents).

Step 4. Abstractive summarization (advanced version)

- Use the *HuggingFace* library (*transformers*) and pre-trained models (for example, *t5-small*, *bart-large-cnn*).
- Generate a coherent abstract for the set of documents.

Step 5. Evaluation of quality

- Evaluate both extractive and abstractive results.
- Evaluate: completeness, coherence, correctness.

Task 3. (25 points) Development and testing of methods for identifying mentions in song lyrics

Tasks:

1. Collect a corpus of song lyrics for at least 4 genres, no fewer than 10 songs per genre.
2. Perform preprocessing: remove stopwords, convert words to lowercase, perform lemmatization.
3. Determine:
   - 10 most frequently occurring words for each genre;
   - words from a specific category (for example: *time of day, seasons, parts of the body,* etc.) for each genre;
   - rare words — words occurring in each genre no more than 3 times.

Data sources
Song lyrics can be taken from open online sources:

- *Genius Lyrics* (https://genius.com/) — English-language song lyrics.
- *AZLyrics* (https://www.azlyrics.com/) — English-language lyrics by genre.
- *LyricFind* (https://www.lyricfind.com/) — licensed lyrics.
- *AmDm.ru* (https://amdm.ru/akkordi/) — Russian-language lyrics.
- *Kaggle datasets:*
  - *Lyrics Dataset* — collected texts by 6 genres.
  - *MetroLyrics dataset* — corpus of songs by genre.

(*It is recommended to use Kaggle, since the data is already structured and labeled by genre.*)

Work procedure
Step 1. Data collection

- Download the dataset from *Kaggle* (for example, *Scrapped Lyrics from 6 Genres*).
- Select 4 genres, 10 songs each.

Step 2. Preprocessing

- Convert text to lowercase.
- Remove punctuation marks.
- Remove stopwords (*NLTK*, *spaCy*).
- Lemmatization (*pymorphy2* for Russian, *spaCy* for English).

Step 3. Frequency analysis

- Count word frequencies in each genre.
- Build the *top-10 words* for each genre.

Step 4. Analysis by categories

- Create dictionaries of categories, for example:
  - *time of day:* morning, day, evening, night;
  - *seasons:* winter, spring, summer, autumn;
  - *parts of the body:* eyes, heart, hands, etc.
- Count the frequency of words from these categories in each genre.

Step 5. Search for rare words

- Determine words that occur in all genres no more than 3 times.

Task 4. (25 points) Language identification

Problem statement
Input: short text (for example, 1–2 sentences).
Output: language of the text (*English, Kazakh, French, Italian,* etc.).

Work procedure
Step 1. Data collection

- Create a small corpus of texts (for example, 20 sentences in 3–4 languages).
- You can take texts from *Wikipedia* or news websites.

Step 2. Preprocessing

- Convert the text to lowercase.
- Remove special symbols and numbers.

Step 3. Features

- Use letter and bigram frequencies.
- You can use *TF-IDF* for words or characters.

Step 4. Model training

- Train a classifier (for example, *Naive Bayes* or *Logistic Regression*).

Step 5. Evaluation of quality

- Split the data into *train/test.*
- Calculate *accuracy.*
- Build a *confusion matrix.*

Control questions

1. What is the difference between extractive and abstractive summarization?
2. Why do methods based on TF-IDF/GraphRank select sentences rather than words?
3. What difficulties arise in the generation of abstractive summaries?
4. What are ROUGE metrics and how are they applied to assess quality?
5. Why is grammatical correctness especially important for abstractive models?
6. What is the difference between lexicon-based and ML-based approaches to sentiment analysis?
7. Why is lemmatization needed?
8. What does the confusion matrix show?
9. Why can accuracy sometimes be misleading?
10. What factors can interfere with the correct determination of sentiment?
11. What is the difference between normalization and standardization of data?
12. What is the advantage of TF-IDF over simple word frequency counts?
13. Why do abstractive models often make grammatical mistakes?
14. How does the ROC curve differ from the Precision-Recall curve?
15. What is the Bag-of-Words method and how does it work?