1. Intro to text cleaning
2. REGEX exercise

Dr. Noa Cohen

Regular Expressions

- A regular expression is a kind of pattern that can be applied to text (Strings, in Python)

- A regular expression either matches the text (or part of the text), or it fails to match it
  - **Which part** of the text matches?
  - **Which parts** of the regular expression match **which parts** of the matched text?
  - Can we **perform substitutions** on the text?

- Regular expressions are an extremely useful tool for manipulating text

Machine Learning and Texts

- Natural language processing (NLP) application examples:
  - Autocorrect – correct to a valid word, correct to a valid word in the context of the sentence
  - Rephrasing of a sentence (see wordtune.com)
  - Recognizing harmful conversations
  - Emails classification (spam \ not spam)
  - Translation between languages

Language Models

- Language models (LM) are an essential element in Natural Language Processing (NLP)

- Many popular NLP applications are using language models
  - Google Assistant, Siri, Amazon's Alexa
  - Text summarization
  - Predicting which word should come next (Google's autofill)
  - Speech recognition
  - Machine translation
  - Optical Character Recognition (OCR)
  - Many more

Language Models

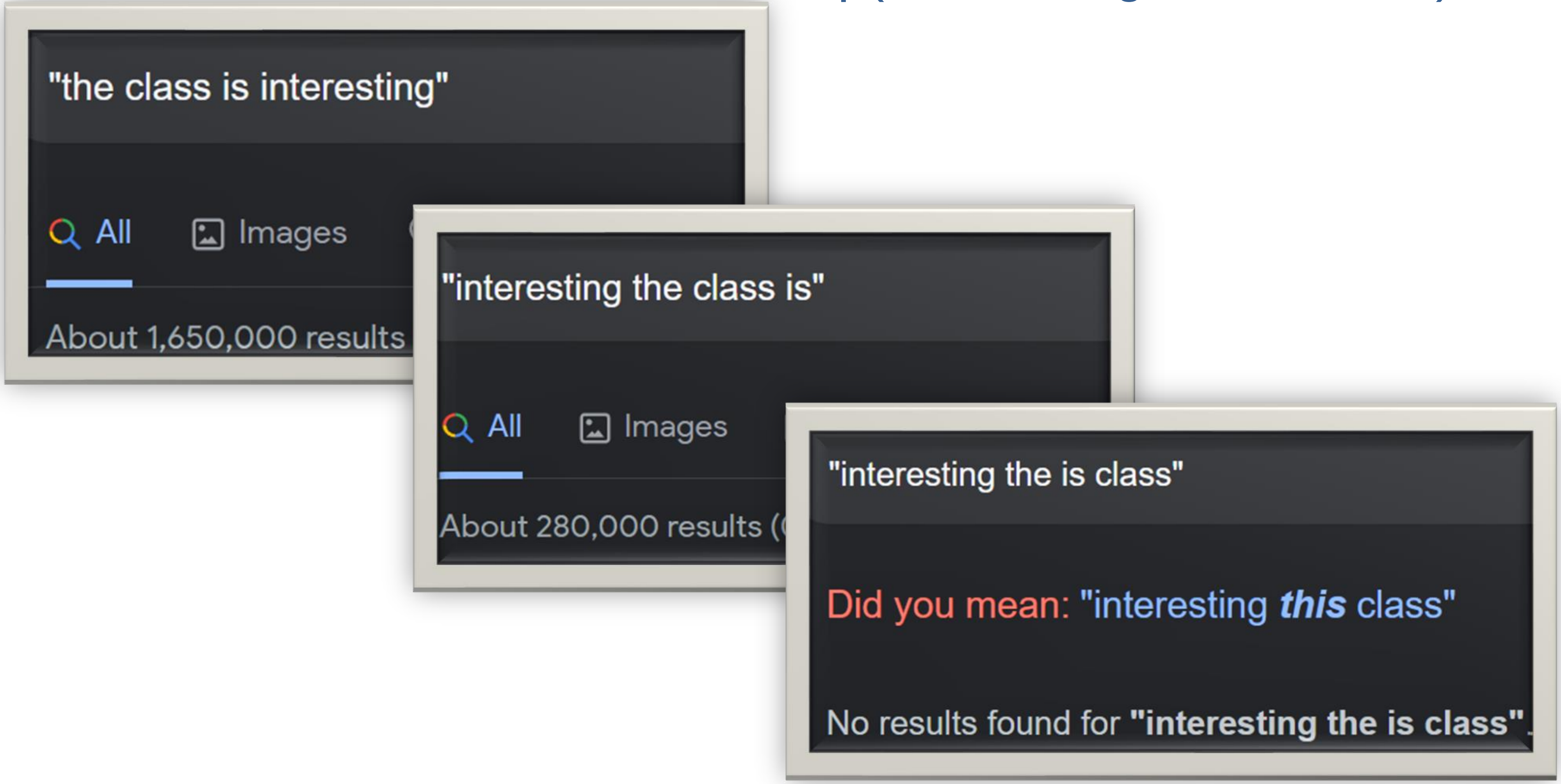
- A language model predicts the probability of a sentence (a sequence of words)

- For example, let us consider Google Translate. In Machine Translation, we convert a set of words from one language to another
  - Often, there are many potential translations
  - Using the probability of each translation to be valid, we can prioritize

p(“the class is interesting”) > p(“interesting the is class”) p("the class is interesting") >

p("interesting the class is") >

p("interesting the is class")



"the class is interesting"

About 1,650,000 results

"interesting the class is"

About 280,000 results (

"interesting the is class"

Did you mean: "interesting ***this*** class"

No results found for **"interesting the is class"**.

Language Models

- In the presented example, by knowing that the probability of the first sentence is higher than the rest, we arrive at the right translation

- This ability to model the rules of a language as a **probability** gives great power for NLP related tasks

Text Cleaning

- When working with texts, we can not build language models or train machine\deep learning models on the raw data

- I.e., we should start be pre-processing of cleaning the text first
  - Ideas for “cleaning the text”?
  - Splitting it into words, handling punctuation, handling case, handling numbers, and more

- There is a set of text preparation methods that we should consider using; the choice of methods depends the specific natural language processing (NLP) task

Text Cleaning - motivation

- Assuming we want to count the number of appearances of the sentence "the class is interesting" in the text

- It could appear as different variations:
  - As a title: "The Class is Interesting" or "THE CLASS IS INTERESTING"
  - With extra spaces: "the   class is   interesting"
  - With a newline: "the class is\ninteresting"
  - With html tags (if we downloaded a web page): "the class is <b>interesting</b>"
  - With punctuation: "the class, is interesting!"

- → Text cleaning is required before we perform NLP analysis

Text Cleaning

- In python we can fetch all punctuations using

```
import string
string.punctuation
```

Exercise - REGEX

- Given a text file (see “TheLittlePrinceRegexExercise.txt”), find all:
  - Appearances of the word I
    - Report how many are found
  - Words starting with a capital letter (e.g., *The*, *Little* , *PRINCE*)
    - Report how many are found
  - Words separated by dashes (e.g., *air-planes*, *grown-up*)
    - Report how many are found in general
    - Keep them in a dictionary with number of appearances per expression (for example: {‘air-planes’: 1, ‘grown-ups’: 6, …})

Exercise - REGEX

- Numbers (e.g., *1943*)
  - Report how many are found
  - Keep a list of these numbers sorted by order of appearances
- Adjacent duplicated words that appear twice (e.g., *and and*)
  - Report how many are found
  - Keep them in a dictionary with number of appearances
  - Print them to a file named “duplications.txt”
- Quotations, i.e., sentences surrounded with “” or ‘’
  - Report how many are found
  - Print them to a file named “quotations.txt”

- Report your answers to the shared file: “REGEX_exercise_shared”

Exercise – Text Cleaning

- Write your own clean_text() function and apply it on the given data (see “TheLittlePrince.txt” in the course’s site)

- Generate the file “TheLittlePrinceCleaned.txt”:
  - Only lower-case letters
  - No punctuations
  - Handle new lines
  - Adjacent duplicated words (could be a mistake in the original text) – replace by one