{"nbformat":4,"nbformat_minor":0,"metadata":{"colab":{"provenance":[],"toc_visible":true,"authorship_tag":"ABX9TyO16yo0Va8ZYxuhPFT5SXKz"},"kernelspec":{"name":"python3","display_name":"Python 3"},"language_info":{"name":"python"}},"cells":[{"cell_type":"markdown","metadata":{"id":"AZmIi46PA_GY"},"source":["# **Build your vocabulary**\n","\n","Building vocabulary from a document will require some string manipulation beyond just the `str.split()`.\n"]},{"cell_type":"markdown","source":["There are certain number of steps that we need to undertake if we want to understand a text pattern or to describe it.\n","1. Finding patterns using **Regular Expressions**\n","2. Then we perform **Text Normalization** (i.e. converting it\n","to a more convenient, standard form) that includes:\n"," 1. **Tokenization**: Separating out individual meaningful words or **tokens** from the running text. \n"," 2. **Lemmatization**: Determining that two words have the same root, despite their surface differences. For example, the words **sang**, **sung**, and **sings** are forms of the verb **sing**. A lemmatizer maps from all of these to sing.\n"," 3. **Stemming**: Stemming efers to a simpler version of lemmatization in which we mainly just strip suffixes from the end of the word.\n"," 4. **Sentence Segmentation**: Breaking up a text into individual sentences, using cues like sentence segmentation periods or exclamation points.\n"],"metadata":{"id":"_p4WB7n1YGRA"}},{"cell_type":"markdown","metadata":{"id":"PVKfPtd7BXuE"},"source":["## **Word tokenization**\n","\n","In NLP, tokenization is a particular kind of document segmentation. \n","\n","Segmentation breaks up text into smaller chunks or segments, with more focused information content.\n","\n","Segmentation can include breaking a document into paragraphs, paragraphs into sentences, sentences into phrases, or phrases into tokens (usually words) and punctuation. \n","\n","Here we focus on segmenting text into tokens, which is called **tokenization**"]},{"cell_type":"markdown","metadata":{"id":"HTHEFNbvBxXm"},"source":["Fundamental building blocks of NLP and their equivalents in a computer language compiler:\n","* tokenizer — scanner, lexer, lexical analyzer\n","* vocabulary — lexicon\n","* parser — compiler\n","* token, term, word, or n-gram — token, symbol, or terminal symbol"]},{"cell_type":"markdown","metadata":{"id":"r1Jtqm86CNJG"},"source":["The simplest way to tokenize a sentence is to use whitespace within a string as the “delimiter” of words. \n","\n","In Python, this can be accomplished with the standard library method `split`, which is available on all `str` object instances as well as on the `str`\n","built-in class itself. "]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"ao-QlrFkCbov","executionInfo":{"status":"ok","timestamp":1685957740389,"user_tz":-330,"elapsed":462,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"754b25cf-d080-48ea-c816-89b99f729aad"},"source":["sentence = \"Thomas Jefferson began building Monticello at the age of 26.\"\n","sentence.split()"],"execution_count":1,"outputs":[{"output_type":"execute_result","data":{"text/plain":["['Thomas',\n"," 'Jefferson',\n"," 'began',\n"," 'building',\n"," 'Monticello',\n"," 'at',\n"," 'the',\n"," 'age',\n"," 'of',\n"," '26.']"]},"metadata":{},"execution_count":1}]},{"cell_type":"markdown","metadata":{"id":"70l8EaRRCg3A"},"source":["**Note:** A good tokenizer should strip off the extra character to create the word “26” as an equivalent class for the words “26,” “26!”, “26?”, and “26.” \n","\n","A more accurate tokenizer would also output a separate token for any sentence-ending punctuation so that a sentence segmenter or sentence boundary detector can find the end of that sentence."]},{"cell_type":"markdown","metadata":{"id":"Km821E0pC59w"},"source":["## **One-hot vectors**\n","\n","**A vocabulary lists all the unique tokens (words) that are present in the text.**\n","\n","We can create a numerical vector representation for **each word**.\n","\n","These vectors are called **one-hot vectors**. \n","\n","A **sequence** of one-hot vectors fully captures the original document text in a sequence of vectors, a table of numbers. \n","\n","That will solve the first problem of NLP, _**turning words into numbers**_:"]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/","height":36},"id":"L-QR7PWvDG3Q","executionInfo":{"status":"ok","timestamp":1685957742104,"user_tz":-330,"elapsed":965,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"bb92664e-cef4-4753-aee3-6ceb16245da5"},"source":["import numpy as np\n","\n","# str.split() is our quick-and-dirty tokenizer\n","token_sequence = str.split(sentence)\n","\n","# Sorted lexicographically (lexically) so numbers come before letters, and capital letters come before lowercase letters.\n","vocab = sorted(set(token_sequence)) \n","\n","', '.join(vocab)"],"execution_count":2,"outputs":[{"output_type":"execute_result","data":{"text/plain":["'26., Jefferson, Monticello, Thomas, age, at, began, building, of, the'"],"application/vnd.google.colaboratory.intrinsic+json":{"type":"string"}},"metadata":{},"execution_count":2}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/","height":36},"id":"zIB4odJIDxUS","executionInfo":{"status":"ok","timestamp":1685957742105,"user_tz":-330,"elapsed":21,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"a97f5c04-8749-4624-f30f-cdeab8c48454"},"source":["num_tokens = len(token_sequence)\n","vocab_size = len(vocab)\n","\n","# The empty table is as wide as our count of unique vocabulary terms and as high as the length of our document, 10 rows by 10 columns.\n","onehot_vectors = np.zeros((num_tokens, vocab_size), int)\n","\n","# For each word in the sentence, mark the column for that word in our vocabulary with a 1.\n","for i, word in enumerate(token_sequence):\n"," onehot_vectors[i, vocab.index(word)] = 1\n","\n","' '.join(vocab)"],"execution_count":3,"outputs":[{"output_type":"execute_result","data":{"text/plain":["'26. Jefferson Monticello Thomas age at began building of the'"],"application/vnd.google.colaboratory.intrinsic+json":{"type":"string"}},"metadata":{},"execution_count":3}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"CJ6qTKJcEPae","executionInfo":{"status":"ok","timestamp":1685957742105,"user_tz":-330,"elapsed":21,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"a77e75d9-f638-4a4a-9f33-fd59ef4431d8"},"source":["onehot_vectors"],"execution_count":4,"outputs":[{"output_type":"execute_result","data":{"text/plain":["array([[0, 0, 0, 1, 0, 0, 0, 0, 0, 0],\n"," [0, 1, 0, 0, 0, 0, 0, 0, 0, 0],\n"," [0, 0, 0, 0, 0, 0, 1, 0, 0, 0],\n"," [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],\n"," [0, 0, 1, 0, 0, 0, 0, 0, 0, 0],\n"," [0, 0, 0, 0, 0, 1, 0, 0, 0, 0],\n"," [0, 0, 0, 0, 0, 0, 0, 0, 0, 1],\n"," [0, 0, 0, 0, 1, 0, 0, 0, 0, 0],\n"," [0, 0, 0, 0, 0, 0, 0, 0, 1, 0],\n"," [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]])"]},"metadata":{},"execution_count":4}]},{"cell_type":"markdown","source":["[0, 0, 0, 1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 1, 0, 0], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 1, 0], [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]]"],"metadata":{"id":"FAoDvmN6gZXL"}},{"cell_type":"markdown","metadata":{"id":"dGAYXoPCESOm"},"source":["A DataFrame keeps track of labels for each column, allowing us to label each column in our table with the token or word it represents. "]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/","height":363},"id":"5xVNGeE4FmZT","executionInfo":{"status":"ok","timestamp":1685957742105,"user_tz":-330,"elapsed":18,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"5bcff6af-6064-4fda-d9b1-da95bea4a274"},"source":["import pandas as pd\n","pd.DataFrame(onehot_vectors, columns=vocab)"],"execution_count":5,"outputs":[{"output_type":"execute_result","data":{"text/plain":[" 26. Jefferson Monticello Thomas age at began building of the\n","0 0 0 0 1 0 0 0 0 0 0\n","1 0 1 0 0 0 0 0 0 0 0\n","2 0 0 0 0 0 0 1 0 0 0\n","3 0 0 0 0 0 0 0 1 0 0\n","4 0 0 1 0 0 0 0 0 0 0\n","5 0 0 0 0 0 1 0 0 0 0\n","6 0 0 0 0 0 0 0 0 0 1\n","7 0 0 0 0 1 0 0 0 0 0\n","8 0 0 0 0 0 0 0 0 1 0\n","9 1 0 0 0 0 0 0 0 0 0"],"text/html":["\n","
\n","
\n","
\n","\n","\n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n","
\n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n","
\n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n","
\n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n","
\n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n","

	26.	Jefferson	Monticello	Thomas	age	at	began	building	of	the
0	0	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	0	0	0
3	0	0	0	0	0	0	0	1	0	0
4	0	0	1	0	0	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	0	0	0	1
7	0	0	0	0	1	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1	0

	26.	Jefferson	Monticello	Thomas	age	at	began	building	of	the
9	1	0	0	0	0	0	0	0	0	0

\n","
\n"," \n"," \n"," \n"," \n"," \n","\n"," \n","
\n","
\n","
\n"," "]},"metadata":{},"execution_count":5}]},{"cell_type":"markdown","metadata":{"id":"Pet5sMY5Fnvz"},"source":["One nice feature of this vector representation of words and tabular representation of documents is that no information is lost. As long as you keep track of which words are indicated by which column, you can reconstruct the original document from this table of one-hot vectors. \n","\n","They're a good choice for any model or NLP pipeline that needs to retain all the\n","meaning inherent in the original text."]},{"cell_type":"markdown","metadata":{"id":"tqQsX-24HBIs"},"source":["In most cases, the vocabulary of tokens you'll use in an NLP pipeline will be millions of tokens. \n","\n","Let's assume you have a million tokens in our NLP pipeline vocabulary. And let's say you have a meager 3,000 books with 3,500 sentences each and 15 words per sentence:"]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"4j5l9kFAHW-B","executionInfo":{"status":"ok","timestamp":1685957742105,"user_tz":-330,"elapsed":17,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"bde108b8-d143-4130-970b-dcac0d9b976f"},"source":["# Number of rows in the table\n","num_rows = 3000 * 3500 * 15\n","num_rows"],"execution_count":6,"outputs":[{"output_type":"execute_result","data":{"text/plain":["157500000"]},"metadata":{},"execution_count":6}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"qDLpaSPSHePv","executionInfo":{"status":"ok","timestamp":1685957742106,"user_tz":-330,"elapsed":16,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"2862a865-dd83-469b-94cc-20f388a6a3a3"},"source":["# Number of bytes, if you use only one byte for each cell in our table\n","num_bytes = num_rows * 1000000\n","num_bytes"],"execution_count":7,"outputs":[{"output_type":"execute_result","data":{"text/plain":["157500000000000"]},"metadata":{},"execution_count":7}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"b3UuWwvsHnxz","executionInfo":{"status":"ok","timestamp":1685957742106,"user_tz":-330,"elapsed":14,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"ddcdac65-0b9d-4d18-ddab-8c571f90cd20"},"source":["num_bytes/1e9 # Gigabytes"],"execution_count":8,"outputs":[{"output_type":"execute_result","data":{"text/plain":["157500.0"]},"metadata":{},"execution_count":8}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"suRxb5VfH1R8","executionInfo":{"status":"ok","timestamp":1685957742106,"user_tz":-330,"elapsed":13,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"871d1484-1726-4e3d-94a4-e5c1fd14c74e"},"source":["# In a python interactive console, the variable name \"_\" is\n","# automatically assigned the value of the previous output. This is handy\n","# if you forget to explicitly assign the output of a function or expression\n","# to a variable name like you did for num_bytes and num_rows.\n","\n","_ / 1000 # Terabytes\n"],"execution_count":9,"outputs":[{"output_type":"execute_result","data":{"text/plain":["157.5"]},"metadata":{},"execution_count":9}]},{"cell_type":"markdown","metadata":{"id":"Jq-P3-HKH6Ey"},"source":["## **Bag-of-words**\n","\n","Storing all those zeros, and trying to remember the order of the words in all our documents, doesn't make much sense. \n","\n","What we want to do is compress the meaning of a document down to its essence. \n","\n","We'd like to compress the document down to a single vector rather than a big table. \n","\n","Let's assume you can ignore the order and grammar of the words, and jumble them all up together into a "bag," one bag for each sentence or short document.\n","\n","A **bag-of-words** vector is useful for summarizing the essence of a document. Even after we sorted all the words lexically, a human can still guess what the sentence was about and so can a machine. You can use this new bag-of-words vector approach to compress the information content for each document into a data structure that's easier to work with.\n","\n","If we summed all the one-hot vectors together we'd get a **bag-of-words** vector. \n","\n","This is also called a **word frequency vector**, because it only counts the frequency of words, not their order. \n","\n","You could use this single vector to represent the whole document or sentence in a single, reasonable length vector. It would only be as long as our vocabulary size (the number of unique tokens you want to keep track of).\n"]},{"cell_type":"markdown","metadata":{"id":"z3UIqPVMJSVq"},"source":["We can also put the tokens into a binary vector indicating the presence or absence of a particular word in a particular sentence. \n","\n","This vector representation of a set of sentences could be "indexed" to indicate which words were used in which document. \n","\n","This index is equivalent to the index you find at the end of many textbooks, except that instead of keeping track of which page a word occurs on, you can keep\n","track of the sentence (or the associated vector) where it occurred. "]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"lviMxInQLrqO","executionInfo":{"status":"ok","timestamp":1685957742107,"user_tz":-330,"elapsed":13,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"c18a4ecc-4b8a-43df-94a2-e2dce4c7963c"},"source":["sentence_bow = {}\n","for token in sentence.split():\n"," sentence_bow[token] = 1\n","\n","sorted(sentence_bow.items())"],"execution_count":10,"outputs":[{"output_type":"execute_result","data":{"text/plain":["[('26.', 1),\n"," ('Jefferson', 1),\n"," ('Monticello', 1),\n"," ('Thomas', 1),\n"," ('age', 1),\n"," ('at', 1),\n"," ('began', 1),\n"," ('building', 1),\n"," ('of', 1),\n"," ('the', 1)]"]},"metadata":{},"execution_count":10}]},{"cell_type":"markdown","metadata":{"id":"Bl1r9o4qLvcI"},"source":["As per the ordering of characters in the ASCII and Unicode character sets, Python's sorted() puts decimal numbers before characters, and capitalized words before lowercase words.\n","\n","Using a dict (or any paired mapping of words to their 0/1 values) to store a binary vector shouldn't waste much space. \n","Using a dictionary to represent our vector ensures that it only has to store a 1 when any one of the possible words in our dictionary appear in a particular document. \n","\n","Because a dictionary "ignores" the absent words, the words labeled with a 0, the dictionary representation only requires a few bytes for each word in our 10-word sentence. \n","And this dictionary could be made even more efficient if you represented each\n","word as an integer pointer to each word's location within our lexicon—the list of\n","words that makes up our vocabulary for a particular application.\n","\n","Let's use an even more efficient form of a dictionary, a Pandas Series. \n","\n","We'll wrap that up in a Pandas DataFrame so we can add more sentences to our\n","binary vector "corpus" of texts about Thomas Jefferson. "]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/","height":81},"id":"aH_pcie7NWY3","executionInfo":{"status":"ok","timestamp":1685957742107,"user_tz":-330,"elapsed":11,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"c153a3c5-d7c3-4825-e72b-eb5bf0cde7c4"},"source":["import pandas as pd\n","\n","df = pd.DataFrame(pd.Series(dict([(token, 1) for token in sentence.split()])), columns=['sent']).T\n","\n","df"],"execution_count":11,"outputs":[{"output_type":"execute_result","data":{"text/plain":[" Thomas Jefferson began building Monticello at the age of 26.\n","sent 1 1 1 1 1 1 1 1 1 1"],"text/html":["\n","
\n","
\n","
\n","\n","\n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n","

	Thomas	Jefferson	began	building	Monticello	at	the	age	of	26.
sent	1	1	1	1	1	1	1	1	1	1

\n","

\n"," \n"," \n"," \n"," \n","\n"," \n","
\n","
\n","
\n"," "]},"metadata":{},"execution_count":11}]},{"cell_type":"markdown","metadata":{"id":"EC-Zpj--Nd-u"},"source":["Let’s add a few more texts to your corpus to see how a DataFrame stacks up."]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/","height":175},"id":"l3nDKUfhNzDo","executionInfo":{"status":"ok","timestamp":1685957743127,"user_tz":-330,"elapsed":1030,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"03bd8cca-94c5-4437-838b-4a43c4b8ce8b"},"source":["sentences = \"\"\"Thomas Jefferson began building Monticello at the age of 26.\\n\"\"\"\n","sentences += \"\"\"Construction was done mostly by local masons and carpenters.\\n\"\"\"\n","sentences += \"He moved into the South Pavilion in 1770.\\n\"\n","sentences += \"\"\"Turning Monticello into a neoclassical masterpiece was Jefferson's obsession.\"\"\"\n","\n","corpus = {}\n","for i, sent in enumerate(sentences.split('\\n')):\n"," corpus['sent{}'.format(i)] = dict((tok, 1) for tok in sent.split())\n","\n","df = pd.DataFrame.from_records(corpus).fillna(0).astype(int).T\n","df[df.columns[:10]]"],"execution_count":12,"outputs":[{"output_type":"execute_result","data":{"text/plain":[" Thomas Jefferson began building Monticello at the age of 26.\n","sent0 1 1 1 1 1 1 1 1 1 1\n","sent1 0 0 0 0 0 0 0 0 0 0\n","sent2 0 0 0 0 0 0 1 0 0 0\n","sent3 0 0 0 0 1 0 0 0 0 0"],"text/html":["\n","
\n","
\n","
\n","\n","\n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n"," \n","

	Thomas	Jefferson	began	building	Monticello	at	the	age	of	26.
sent0	1	1	1	1	1	1	1	1	1	1
sent1	0	0	0	0	0	0	0	0	0	0
sent2	0	0	0	0	0	0	1	0	0	0
sent3	0	0	0	0	1	0	0	0	0	0

\n","
\n"," \n"," \n"," \n"," \n","\n"," \n","
\n","
\n","
\n"," "]},"metadata":{},"execution_count":12}]},{"cell_type":"markdown","metadata":{"id":"o1Njt7iaN36g"},"source":["With a quick scan, you can see little overlap in word usage for these sentences. Among the first seven words in your vocabulary, only the word “Monticello” appears in more than one sentence. \n","\n","Now you need to be able to compute this overlap within your pipeline whenever you want to compare documents or search for similar documents. \n","\n","One way to check for the similarities between sentences is to count the number of overlapping tokens using a dot product."]},{"cell_type":"markdown","metadata":{"id":"p9nnMGVCO2xu"},"source":["## **Dot product**\n","\n","The scalar value output by the scalar product can be calculated by multiplying all the elements of one vector by all the elements of a second vector, and then adding up those normal multiplication products.\n","\n"]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"NSA3A-04QWM1","executionInfo":{"status":"ok","timestamp":1685957743128,"user_tz":-330,"elapsed":28,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"fd92b7de-9107-4b01-863f-552405b7666e"},"source":["v1 = np.array([1, 2, 3])\n","v2 = np.array([2, 3, 4])\n","v1.dot(v2)"],"execution_count":13,"outputs":[{"output_type":"execute_result","data":{"text/plain":["20"]},"metadata":{},"execution_count":13}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"gOaA__2cQZtV","executionInfo":{"status":"ok","timestamp":1685957743128,"user_tz":-330,"elapsed":26,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"afff7180-1720-4482-8b17-19194aa4b00e"},"source":["# Multiplication of numpy arrays is a “vectorized” operation that is very efficient.\n","(v1 * v2).sum()"],"execution_count":14,"outputs":[{"output_type":"execute_result","data":{"text/plain":["20"]},"metadata":{},"execution_count":14}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"nXR__l63Ql4c","executionInfo":{"status":"ok","timestamp":1685957743128,"user_tz":-330,"elapsed":24,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"e67beffa-f259-4e85-919b-7064d771aaf8"},"source":["# You shouldn’t iterate through vectors this way unless you want to slow down your pipeline.\n","sum([x1 * x2 for x1, x2 in zip(v1, v2)])"],"execution_count":15,"outputs":[{"output_type":"execute_result","data":{"text/plain":["20"]},"metadata":{},"execution_count":15}]},{"cell_type":"markdown","metadata":{"id":"bjVYrB5hQmjV"},"source":["### **Measuring bag-of-words overlap**\n","\n","If we can measure the bag of words overlap for two vectors, we can get a good estimate\n","of how similar they are in the words they use. "]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"XGBkm8wJRLrl","executionInfo":{"status":"ok","timestamp":1685957743128,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"78ace0d6-f546-4024-c938-2f6a2d96f17c"},"source":["df = df.T\n","df.sent0.dot(df.sent1)"],"execution_count":16,"outputs":[{"output_type":"execute_result","data":{"text/plain":["0"]},"metadata":{},"execution_count":16}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"iKW9UngORMNW","executionInfo":{"status":"ok","timestamp":1685957743129,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"6dc2c432-2542-4c2f-ba92-b46237b84a54"},"source":["df.sent0.dot(df.sent2)"],"execution_count":17,"outputs":[{"output_type":"execute_result","data":{"text/plain":["1"]},"metadata":{},"execution_count":17}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"B-JQwpzORc6s","executionInfo":{"status":"ok","timestamp":1685957743129,"user_tz":-330,"elapsed":18,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"f39b1fd7-894c-440c-c3bc-939bf8769eab"},"source":["df.sent0.dot(df.sent3)"],"execution_count":18,"outputs":[{"output_type":"execute_result","data":{"text/plain":["1"]},"metadata":{},"execution_count":18}]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"1a-R7RlfRepn","executionInfo":{"status":"ok","timestamp":1685957743129,"user_tz":-330,"elapsed":15,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}},"outputId":"0bbce7b1-f153-49fa-ad8e-e515f9d7f2a9"},"source":["# to find the word that is shared by

```
sent0 and sent3, the word that gave you \n","# that last dot product of 1:\n","[(k, v) for (k, v) in (df.sent0 & df.sent3).items() if
v]"],"execution_count":19,"outputs":[{"output_type":"execute_result","data":{"text/plain":["[('Monticello', 1)]"]},"metadata":{},"execution_count":19}]},
{"cell_type":"markdown","metadata":{"id":"7vXwpAclSIDI"},"source":["This is your first vector space model (VSM) of natural language documents
(sentences). Not only are dot products possible, but other vector operations are defined for these bag-of-word vectors: addition, subtraction, OR,
AND, and so on. \n","\n","You can even compute things such as Euclidean distance or the angle between these vectors. This representation of a
document as a binary vector has a lot of power. It was a mainstay for document retrieval and search for many years."]},
{"cell_type":"markdown","metadata":{"id":"M4LyQP1nSYjr"},"source":["## **Using Regular Expressions**\n","\n"]},{"cell_type":"code","metadata":
{"colab":{"base_uri":"https://localhost:8080/"},"id":"-XiwabE-TFgT","executionInfo":
{"status":"ok","timestamp":1685957743129,"user_tz":-330,"elapsed":13,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"94c38543-e76b-4a70-9b3c-7cfaeb1b1e75"},"source":["import re\n","sentence =
\"\"\"Thomas Jefferson began building Monticello at the age of 26.\"\"\"\n","\n","# This splits the sentence on whitespace or punctuation that occurs
at least\n","# once (note the '+' after the closing square bracket in the regular expression).\n","tokens = re.split(r'[-\\s.,;!?]+',
sentence)\n","tokens"],"execution_count":20,"outputs":[{"output_type":"execute_result","data":{"text/plain":["['Thomas',\n"," 'Jefferson',\n","
'began',\n"," 'building',\n"," 'Monticello',\n"," 'at',\n"," 'the',\n"," 'age',\n"," 'of',\n"," '26',\n"," '']"]},"metadata":{},"execution_count":20}]},
{"cell_type":"markdown","metadata":{"id":"Muf4bJ2CTRM6"},"source":["* The square brackets ([ and ]) are used to indicate a character class, a
set of characters. \n","\n","* The plus sign after the closing square bracket (]) means that a match must contain one or more of the
characters\n","inside the square brackets. \n","* The \\s within the character class is a shortcut to a predefined character class that includes all
whitespace characters like those created when\n","you press the [space], [tab], and [return] keys. \n","* The character class r'[\\s]' is equivalent to
r' \\t\\n\\r\\x0b\\x0c'. \n","* The six whitespace characters are space (' '), tab ('\\t'), return ('\\r'), newline ('\\n'), and form-feed ('\\f').\n","* A character
range is a special kind of character class indicated within square brackets and a hyphen, like r'[a-z]' to match all lowercase letters. \n","* The
character range r'[0-9]' matches any digit 0 through 9 and is equivalent to r'[0123456789]'. \n","* The regular expression r'[\\_a-zA-Z]' would match
any underscore character ('\\_') or letter of the English alphabet (upper or lowercase).\n","* You can’t put a hyphen just anywhere inside your
square brackets, because\n","the regex parser may think you mean a character range like r'[0-9]'. To let it know that you really mean a literal
hyphen character, you have to put it right after the open square bracket for the character class. \n","\n","\n","The re.split function goes through
each character in the input string (the second argument, sentence) left to right looking for any matches based on the “program” in the regular
expression (the first argument, r'[-\\s.,;!?]+'). \n","\n","When it finds a match, it breaks the string right before that matched character and right after
it, skipping over the matched character or characters. \n","\n","So the `re.split` line will work just like `str.split`, but it will work for any kind of
character or multicharacter sequence that matches your regular expression.\n","\n","The parentheses (\"(\" and \")\") are used to group regular
expressions just like\n","they’re used to group mathematical, Python, and most other programming language\n","expressions. These parentheses
force the regular expression to match the entire\n","expression within the parentheses before moving on to try to match the characters that follow
the parentheses."]},{"cell_type":"markdown","metadata":{"id":"icALIKfPVLBq"},"source":["The regular expression module in Python allows you to
precompile regular expressions,a which you then can reuse across your code base. \n","For example, you might have a regex that extracts
phone numbers."]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"ba3VFT7dWxyJ","executionInfo":
{"status":"ok","timestamp":1685957743130,"user_tz":-330,"elapsed":13,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"53e5729f-9f92-41c4-fc50-c6c68ea8635e"},"source":["pattern = re.compile(r\"([-
\\s.,;!?])+\")\n","tokens = pattern.split(sentence)\n","tokens[-10:] # just the last 10 tokens"],"execution_count":21,"outputs":
[{"output_type":"execute_result","data":{"text/plain":["[' ', 'the', ' ', 'age', ' ', 'of', ' ', '26', '.', '']"]},"metadata":{},"execution_count":21}]},
{"cell_type":"markdown","metadata":{"id":"5OJ3WhJlW2hn"},"source":["This simple regular expression is helping to split off the period from the
end of the token “26.” \n","\n","However, you have a new problem. You need to filter the whitespace and punctuation characters you don’t want to
include in your vocabulary."]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"d_zwD6YIXaWt","executionInfo":
{"status":"ok","timestamp":1685957743130,"user_tz":-330,"elapsed":11,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"a192d238-acdf-4d00-e5bd-0a574a569083"},"source":["sentence = \"Thomas Jefferson
began building Monticello at the age of 26.\"\n","tokens = pattern.split(sentence)\n","[x for x in tokens if x and x not in '-
\\t\\n.,;!?']"],"execution_count":22,"outputs":[{"output_type":"execute_result","data":{"text/plain":["['Thomas',\n"," 'Jefferson',\n"," 'began',\n","
'building',\n"," 'Monticello',\n"," 'at',\n"," 'the',\n"," 'age',\n"," 'of',\n"," '26']"]},"metadata":{},"execution_count":22}]},{"cell_type":"code","metadata":
{"colab":{"base_uri":"https://localhost:8080/"},"id":"LcGG0sPfXI14","executionInfo":
{"status":"ok","timestamp":1685957743130,"user_tz":-330,"elapsed":10,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"bcd32595-ab74-4627-c3d8-e0f7b4832ba6"},"source":["# Another waay of doing the same
thing:-\n","# If you want practice with lambda and filter(), use\n","\n","list(filter(lambda x: x if x and x not in '- \\t\\n.,;!?' else None,
tokens))"],"execution_count":23,"outputs":[{"output_type":"execute_result","data":{"text/plain":["['Thomas',\n"," 'Jefferson',\n"," 'began',\n","
'building',\n"," 'Monticello',\n"," 'at',\n"," 'the',\n"," 'age',\n"," 'of',\n"," '26']"]},"metadata":{},"execution_count":23}]},
{"cell_type":"markdown","metadata":{"id":"7igkjNeNX6os"},"source":["![image.png]
(data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAvUAAAG1CAYAAAB58gt8AAAgAElEQVR4Aexd91sT2ff+/jMJCU3FFQvsx65gQbGy6l
{"cell_type":"markdown","metadata":{"id":"Gc4CyGOwYHpb"},"source":["As you can imagine, tokenizers can easily become complex. In one
case, you might\n","want to split based on periods, but only if the period isn’t followed by a number, in order to avoid splitting decimals. In another
case, you might not want to split after a period that is part of “smiley” emoticon symbol, such as in a Twitter message.\n","\n","Several Python
libraries implement tokenizers, each with its own advantages and\n","disadvantages:\n","* **spaCy**—Accurate , flexible, fast, Python\n","*
**Stanford CoreNLP**—More accurate, less flexible, fast, depends on Java 8\n","* **NLTK**—Standard used by many NLP contests and
comparisons, popular, Python. \n","\n","NLTK and Stanford CoreNLP have been around the longest and are the most widely\n","used for
comparison of NLP algorithms in academic papers. \n","\n","Even though the Stanford CoreNLP has a Python API, it relies on the Java 8
CoreNLP backend, which must be installed and configured separately. \n","\n","So you can use the Natural Language Toolkit (NLTK) tokenizer
here to get you up and running quickly; it will help you duplicate the results you see in academic papers and blog posts"]},
{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"zOWqrmoZYlvn","executionInfo":
{"status":"ok","timestamp":1685957744266,"user_tz":-330,"elapsed":1144,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"77e442ca-b566-4ff7-a38e-0301d0aa5ed4"},"source":["from nltk.tokenize import
RegexpTokenizer\n","tokenizer = RegexpTokenizer(r'\\w+|$[0-9.]+|\\S+')\n","tokenizer.tokenize(sentence)"],"execution_count":24,"outputs":
[{"output_type":"execute_result","data":{"text/plain":["['Thomas',\n"," 'Jefferson',\n"," 'began',\n"," 'building',\n"," 'Monticello',\n"," 'at',\n"," 'the',\n","
'age',\n"," 'of',\n"," '26',\n"," '.']"]},"metadata":{},"execution_count":24}]},{"cell_type":"markdown","metadata":{"id":"xzBKKv-AYsEN"},"source":["This
tokenizer ignores whitespace tokens. It also separates sentence-ending trailing punctuation from tokens that do not contain any other punctuation
characters.\n","\n","An even better tokenizer is the **[Treebank Word Tokenizer](http://www.nltk.org/api/nltk.tokenize.html#module-
nltk.tokenize.treebank)** from the **NLTK** package. It incorporates a variety of common rules for English word tokenization. For example, it
separates phrase-terminating punctuation (?!.;,) from adjacent tokens and retains decimal numbers containing a period as a single token.
\n","\n","In addition it contains rules for English contractions. For example “don’t” is tokenized as [\"do\", \"n’t\"]. This tokenization will help with
subsequent steps in the NLP pipeline, such as stemming. "]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"NUoL7Os4ZOJX","executionInfo":
{"status":"ok","timestamp":1685957744267,"user_tz":-330,"elapsed":28,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"ee7640ee-822b-4414-a75e-2e888f84195d"},"source":["from nltk.tokenize import
TreebankWordTokenizer\n","sentence = \"\"\"Monticello wasn't designated as UNESCO World Heritage Site until 1987.\"\"\"\n","tokenizer =
TreebankWordTokenizer()\n","tokenizer.tokenize(sentence)"],"execution_count":25,"outputs":[{"output_type":"execute_result","data":{"text/plain":["
['Monticello',\n"," 'was',\n"," \"n't\",\n"," 'designated',\n"," 'as',\n"," 'UNESCO',\n"," 'World',\n"," 'Heritage',\n"," 'Site',\n"," 'until',\n"," '1987',\n","
'.']"]},"metadata":{},"execution_count":25}]},{"cell_type":"markdown","metadata":{"id":"2r69c9aJZcNO"},"source":["## **n-grams**\n","\n","An **n-
gram** is a sequence containing up to n elements that have been extracted from a sequence of those elements, usually a string. \n","> *The
“elements” of an n-gram can be characters, syllables, words, or even symbols like “A,” “T,” “G,” and “C” used to represent a DNA
sequence*\n","\n","**Why bother with n-grams?**\n","\n","> *n-grams retain a context of a word when we tie it to its neighbor(s) in the NLP
pipeline*\n","\n","When a sequence of tokens is vectorized into a bag-of-words vector, it loses a lot of the meaning inherent in the order of those
```

words. By extending the concept of a token to include multiword tokens, n-grams, the NLP pipeline can retain much of the meaning inherent in the order of words in the statement. \n","\n","For example, the meaning-inverting word “not” will remain attached to its neighboring words, where it belongs. Without n-gram tokenization, it would be free floating. Its meaning would be associated with the entire sentence or document rather than its neighboring words. The 2-gram “was not” retains much more of the meaning of the individual words “not” and “was” than those 1-grams alone in a bag-of-words vector. "]},{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"4uA4cbkQ0hmC","executionInfo":
{"status":"ok","timestamp":1685957744267,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"2624d607-5702-418b-d46c-b41c203a80dd"},"source":["# tokenize_2grams(\"Thomas Jefferson began building Monticello at the age of 26.\")\n","sentence = \"\"\"Thomas Jefferson began building Monticello at the age of
26.\"\"\"\n","pattern = re.compile(r\"([-\\s.,;!?])+\")\n","tokens = pattern.split(sentence)\n","tokens = [x for x in tokens if x and x not in '-
\\t\\n.,;!?']\n","tokens"],"execution_count":26,"outputs":[{"output_type":"execute_result","data":{"text/plain":["['Thomas',\n"," 'Jefferson',\n","
'began',\n"," 'building',\n"," 'Monticello',\n"," 'at',\n"," 'the',\n"," 'age',\n"," 'of',\n"," '26']"]},"metadata":{},"execution_count":26}]},
{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"RnnhAKPl0iyH","executionInfo":
{"status":"ok","timestamp":1685957744267,"user_tz":-330,"elapsed":20,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"df3a9bdc-5451-40d6-d516-c0056370c23d"},"source":["from nltk.util import
ngrams\n","list(ngrams(tokens, 2))"],"execution_count":27,"outputs":[{"output_type":"execute_result","data":{"text/plain":["[('Thomas',
'Jefferson'),\n"," ('Jefferson', 'began'),\n"," ('began', 'building'),\n"," ('building', 'Monticello'),\n"," ('Monticello', 'at'),\n"," ('at', 'the'),\n"," ('the',
'age'),\n"," ('age', 'of'),\n"," ('of', '26')]"]},"metadata":{},"execution_count":27}]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"xo2F56QG1Ryg","executionInfo":
{"status":"ok","timestamp":1685957744267,"user_tz":-330,"elapsed":17,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"63de8ff9-8204-4fe5-fcce-11109c00c611"},"source":["list(ngrams(tokens,
3))"],"execution_count":28,"outputs":[{"output_type":"execute_result","data":{"text/plain":["[('Thomas', 'Jefferson', 'began'),\n"," ('Jefferson', 'began',
'building'),\n"," ('began', 'building', 'Monticello'),\n"," ('building', 'Monticello', 'at'),\n"," ('Monticello', 'at', 'the'),\n"," ('at', 'the', 'age'),\n"," ('the', 'age',
'of'),\n"," ('age', 'of', '26')]"]},"metadata":{},"execution_count":28}]},{"cell_type":"markdown","metadata":{"id":"JXI4WLNE1ZTI"},"source":["## **Stop
Words**\n","\n","Stop words are common words in any language that occur with a high frequency but carry much less substantive information about the meaning of a phrase.\n","* a, an\n","* the, this\n","* and, or\n","* of, on\n","\n","Historically stop words have been excluded from NLP pipelines in order to reduce\n","the computational effort to extract information from a text. \n","\n","Even though the words themselves carry little information, the stop words can provide important relational information as part of an n-gram.\n","\n","* Mark reported to the CEO\n","* Suzanne reported as the CEO to the board\n","\n","In the NLP pipeline, we might create 4-grams such as _`reported to the CEO`_ and *`reported as the CEO`*. \n","\n","If you remove the stop words from the 4-grams, both examples would be reduced to _`reported CEO`_\n","\n","Retaining the stop words within your pipeline creates another problem:\n","* it increases the length of the n-grams required to make use of these connections formed by the otherwise meaningless stop words. \n"]},{"cell_type":"markdown","metadata":{"id":"yfI0bZkA9qDD"},"source":["If you do decide to arbitrarily filter out a set of stop words during tokenization, a\n","Python list comprehension is sufficient."]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"xjFEbV6LFOTp","executionInfo":
{"status":"ok","timestamp":1685957744268,"user_tz":-330,"elapsed":16,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"f6c0ab7d-d4ce-457d-e5b5-37fb1a93bfde"},"source":["stop_words = ['a', 'an', 'the', 'on', 'of',
'off', 'this', 'is']\n","tokens = ['the', 'house', 'is', 'on', 'fire']\n","tokens_without_stopwords = [x for x in tokens if x not in
stop_words]\n","print(tokens_without_stopwords)"],"execution_count":29,"outputs":[{"output_type":"stream","name":"stdout","text":["['house',
'fire']\n"]}]},{"cell_type":"markdown","metadata":{"id":"LngYPMAzFV45"},"source":["NLTK provides a complete list of “canonical” stop words:"]},
{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"SXBz6aWFF54n","executionInfo":
{"status":"ok","timestamp":1685957744268,"user_tz":-330,"elapsed":14,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"5fe1346f-3aa9-48ce-99b2-e3ab7d1b9b18"},"source":["import
nltk\n","nltk.download('stopwords')\n","stop_words = nltk.corpus.stopwords.words('english')\n","len(stop_words)\n","#
stop_words[:7]"],"execution_count":30,"outputs":[{"output_type":"stream","name":"stderr","text":["[nltk_data] Downloading package stopwords to
/root/nltk_data...\n","[nltk_data] Unzipping corpora/stopwords.zip.\n"]},{"output_type":"execute_result","data":{"text/plain":["179"]},"metadata":
{},"execution_count":30}]},{"cell_type":"markdown","metadata":{"id":"-M_3-j9AGApg"},"source":["**WARNING** The set of English stop words that sklearn uses is quite different from those in NLTK."]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"4EIpFXVsHty9","executionInfo":
{"status":"ok","timestamp":1685957744268,"user_tz":-330,"elapsed":13,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"3b80fc32-af51-43f1-cb81-bc041de7625a"},"source":["from sklearn.feature_extraction.text import ENGLISH_STOP_WORDS as sklearn_stop_words\n","\n","len(sklearn_stop_words)"],"execution_count":31,"outputs":
[{"output_type":"execute_result","data":{"text/plain":["318"]},"metadata":{},"execution_count":31}]},{"cell_type":"markdown","metadata":
{"id":"Rs1X0BKQH0iu"},"source":["## **Normalizing your vocabulary**\n","\n","Another vocabulary reduction technique is to normalize the vocabulary so that tokens that mean similar things are combined into a single, normalized form. \n","\n","Doing so reduces the number of tokens you need to retain in your vocabulary and also improves the association of meaning across those different “spellings” of a token or ngram in the corpus.\n","\n","**Case Folding**\n","\n","> Case folding is when you consolidate multiple “spellings” of a word that differ only in their capitalization\n","\n","Capitalization is often used to indicate that a word is a proper noun, the name of a person, place, or thing.\n","\n"]},
{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"YZv84wouPwLD","executionInfo":
{"status":"ok","timestamp":1685957744269,"user_tz":-330,"elapsed":12,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"80ed94fc-058c-4b80-b74a-9e8839bcdf9c"},"source":["tokens = ['House', 'Visitor',
'Center']\n","normalized_tokens = [x.lower() for x in tokens]\n","print(normalized_tokens)"],"execution_count":32,"outputs":
[{"output_type":"stream","name":"stdout","text":["['house', 'visitor', 'center']\n"]}]},{"cell_type":"markdown","metadata":
{"id":"moH3rQolPyZD"},"source":["**Stemming**\n","\n","Another common vocabulary normalization technique is to eliminate the small meaning differences of pluralization or possessive endings of words, or even various verb forms. \n","\n","This normalization, identifying a common stem among various forms of a\n","word, is called stemming. \n","> E.g. the words housing and houses share the same stem, house.
\n","\n","Stemming removes suffixes from words in an attempt to combine words\n","with similar meanings together under their common stem.
\n","\n","A stem isn’t required to be a properly spelled word, but merely a token, or label, representing several possible spellings of a word.\n","\n","However, stemming could greatly reduce the “precision” score for your search engine, because it might return many more irrelevant documents along with the relevant ones."]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/","height":36},"id":"msjBUzG4QqkP","executionInfo":
{"status":"ok","timestamp":1685957744269,"user_tz":-330,"elapsed":9,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"e39e578e-326b-4001-9d73-9e409ed08231"},"source":["# Here’s a simple stemmer implementation in pure Python that \n","# can handle trailing S’s:\n","import re\n","def stem(phrase):\n"," return ' '.join([re.findall('^(.*ss|.*?)(s)?$', word)[0][0].strip(\"'\") for word in phrase.lower().split()])\n","\n","stem('houses')\n"],"execution_count":33,"outputs":
[{"output_type":"execute_result","data":{"text/plain":["'house'"],"application/vnd.google.colaboratory.intrinsic+json":{"type":"string"}},"metadata":
{},"execution_count":33}]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/","height":36},"id":"b9IWmRbzQzLw","executionInfo":
{"status":"ok","timestamp":1685957744269,"user_tz":-330,"elapsed":8,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"d2aa53b2-0fd5-400f-e90a-e2e988e91375"},"source":["stem(\"Doctor House's
calls\")"],"execution_count":34,"outputs":[{"output_type":"execute_result","data":{"text/plain":["'doctor house
call'"],"application/vnd.google.colaboratory.intrinsic+json":{"type":"string"}},"metadata":{},"execution_count":34}]},
{"cell_type":"markdown","metadata":{"id":"JljjCRGlRDez"},"source":["Two of the most popular stemming algorithms are \n","1. Porter stemmers
\n","2. Snowball stemmers\n","\n","The Porter stemmer is named for the computer scientist Martin Porter. Porter\n","is also responsible for enhancing the Porter stemmer to create the Snowball stemmer.Porter dedicated much of his lengthy career to documenting and improving stemmers,\n","due to their value in information retrieval (keyword search). These stemmers implement more complex rules than our simple regular expression. This enables the stemmer to handle the complexities of English spelling and word ending rules:"]},

```
{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/","height":36},"id":"F8lR7m8LRrcj","executionInfo":
{"status":"ok","timestamp":1685957744869,"user_tz":-330,"elapsed":30,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"117b9ae4-719a-4068-fa42-2e7e7ad01763"},"source":["from nltk.stem.porter import
PorterStemmer\n","\n","stemmer = PorterStemmer()\n","' '.join([stemmer.stem(w).strip(\"'\") for w in \"dish washer's washed
dishes\".split()])"],"execution_count":35,"outputs":[{"output_type":"execute_result","data":{"text/plain":["'dish washer wash
dish'"],"application/vnd.google.colaboratory.intrinsic+json":{"type":"string"}},"metadata":{},"execution_count":35}]},
{"cell_type":"markdown","metadata":{"id":"CR6HphOpRxYK"},"source":["![image.png]
(data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAA4MAAANXCAYAAACPOIrDAAAgAElEQVR4Aey9V1RT29vv/xvnfe/Oufmf23e87znj/E9
{"cell_type":"markdown","metadata":{"id":"PfOTYyY2SUHQ"},"source":["**Lemmatization**\n","\n","If you have access to information about
connections between the meanings of various words, you might be able to associate several words together even if their spelling is quite
different. \n","\n","This more extensive normalization down to the semantic root of a word—its lemma—is called lemmatization.\n","\n","Any NLP
pipeline that wants to “react” the same for multiple different spellings of the same basic root word can benefit from a lemmatizer. \n","\n","NLTK
package provides functions for identifying word lemmas. You must tell the WordNetLemmatizer which part of speech your are interested in, if you
want to find the most accurate lemma:"]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"Q1EvHOd3TqFT","executionInfo":
{"status":"ok","timestamp":1685957744870,"user_tz":-330,"elapsed":27,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"d510c28a-c162-4aef-8066-96b3bedfa507"},"source":["nltk.download('wordnet')\n","#
nltk.download()\n","\n","from nltk.stem import WordNetLemmatizer\n","lemmatizer = WordNetLemmatizer()"],"execution_count":36,"outputs":
[{"output_type":"stream","name":"stderr","text":["[nltk_data] Downloading package wordnet to /root/nltk_data...\n"]}]},{"cell_type":"code","source":
["# from nltk.stem import WordNetLemmatizer\n","\n","# lemmatizer = WordNetLemmatizer()\n","# lemmatizer.lemmatize(\"better\")"],"metadata":
{"id":"TvwqID6G4FxD","executionInfo":{"status":"ok","timestamp":1685957744870,"user_tz":-330,"elapsed":19,"user":{"displayName":"Dr.
Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":37,"outputs":[]},{"cell_type":"code","metadata":
{"id":"Diq9wrLYT05J","executionInfo":{"status":"ok","timestamp":1685957744870,"user_tz":-330,"elapsed":18,"user":{"displayName":"Dr.
Muneendra Ojha","userId":"06154531826228794240"}}},"source":["# # The default part of speech is “n” for noun.\n","#
print(lemmatizer.lemmatize(\"better\"))\n","# # 'better'\n","\n","# # “a” indicates the adjective part of speech.\n","#
print(lemmatizer.lemmatize(\"better\", pos=\"a\"))\n","# # 'good'\n","# print(lemmatizer.lemmatize(\"good\", pos=\"a\"))\n","# # 'good'\n","#
print(lemmatizer.lemmatize(\"goods\", pos=\"a\"))\n","# # 'goods'\n","# print(lemmatizer.lemmatize(\"goods\", pos=\"n\"))\n","# # 'good'\n","#
print(lemmatizer.lemmatize(\"goodness\", pos=\"n\"))\n","# # 'goodness'\n","# print(lemmatizer.lemmatize(\"best\", pos=\"a\"))\n","# #
'best'"],"execution_count":38,"outputs":[]},{"cell_type":"markdown","metadata":{"id":"wsx15ZnOUDkv"},"source":["the NLTK lemmatizer is restricted
to the connections within the Princeton WordNet graph of word meanings. So the word “best” doesn’t lemmatize to the same root as “better.” This
graph is also missing the connection between “goodness” and “good.” \n","\n","A Porter stemmer, on the other hand, would make this connection
by blindly stripping off the “ness” ending of all words:"]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/","height":36},"id":"PThmNSYVU0ZB","executionInfo":
{"status":"ok","timestamp":1685957744870,"user_tz":-330,"elapsed":17,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"7c41bdca-db8d-4d25-f467-6308540d043c"},"source":
["stemmer.stem('goodness')"],"execution_count":39,"outputs":[{"output_type":"execute_result","data":{"text/plain":
["'good'"],"application/vnd.google.colaboratory.intrinsic+json":{"type":"string"}},"metadata":{},"execution_count":39}]},
{"cell_type":"markdown","metadata":{"id":"Hbsk-54IU0p7"},"source":["**Use Cases**\n","\n","When should you use a lemmatizer or a stemmer?
Stemmers are generally faster to\n","compute and require less-complex code and datasets. But stemmers will make more\n","errors and stem a
far greater number of words, reducing the information content or\n","meaning of your text much more than a lemmatizer would. Both stemmers
and lemmatizers will reduce your vocabulary size and increase the ambiguity of the text. But\n","lemmatizers do a better job retaining as much of
the information content as possible\n","based on how the word was used within the text and its intended meaning. Therefore,\n","some NLP
packages, such as spaCy, don’t provide stemming functions and only offer\n","lemmatization methods."]},{"cell_type":"markdown","metadata":
{"id":"CAQdYeIEVsNT"},"source":["# **Measuring Sentiment**\n","\n","_**Sentiment analysis**_ — measuring the sentiment of phrases or chunks
of text—is a common application of NLP. \n","\n","In many companies it’s the main thing an NLP engineer is asked to do. Companies like to know
what users think of their products.\n","\n","**Question**: What kind of pipeline would you create to measure the sentiment of a block of text?"]},
{"cell_type":"markdown","metadata":{"id":"R9lcvRYjVtpr"},"source":["There are two approaches to sentiment analysis:\n","* A rule-based algorithm
composed by a human\n","* A machine learning model learned from data by a machine"]},{"cell_type":"markdown","metadata":{"id":"d-
Wu3rOZZJww"},"source":["## **VADER—A rule-based sentiment analyzer**\n","\n","Hutto and Gilbert at Georgia Tech created a rule-based
sentiment analysis algorithms called **VADER** (**V**alence **A**ware **D**ictionary for s**E**ntiment **R**easoning).\n","\n","The NLTK
package has an implementation of the VADER algorithm in `nltk.sentiment.vader`. Hutto himself maintains the Python package
`vaderSentiment`."]},{"cell_type":"markdown","metadata":{"id":"9d9LKlyYbpqJ"},"source":["The first approach to sentiment analysis uses human-
designed rules, sometimes called heuristics, to measure sentiment. \n","\n","A common rule-based approach to sentiment analysis is to find
keywords in the text and map each one to numerical scores or weights in a dictionary or “mapping”—a Python dict, for example. \n","\n","You can
use stems, lemmas, or n-gram tokens in your dictionary, rather\n","than just words. \n","\n","The “rule” in your algorithm would be to add up these
scores for each keyword in a document that you can find in your dictionary of sentiment scores. \n","\n","Of course you need to hand-compose
this dictionary of keywords and their sentiment scores before you can run this algorithm on a body of text. "]},{"cell_type":"code","metadata":
{"colab":{"base_uri":"https://localhost:8080/"},"id":"5QA8L78uaoBE","executionInfo":
{"status":"ok","timestamp":1685957749070,"user_tz":-330,"elapsed":4214,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"a294a70a-4c98-443b-89b6-e8a3205b8061"},"source":["pip install
vaderSentiment"],"execution_count":40,"outputs":[{"output_type":"stream","name":"stdout","text":["Looking in indexes: https://pypi.org/simple,
https://us-python.pkg.dev/colab-wheels/public/simple/\n","Collecting vaderSentiment\n"," Downloading vaderSentiment-3.3.2-py2.py3-none-
any.whl (125 kB)\n","\u001b[2K \u001b[90m━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━\u001b[0m
\u001b[32m126.0/126.0 kB\u001b[0m \u001b[31m6.5 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hRequirement already
satisfied: requests in /usr/local/lib/python3.10/dist-packages (from vaderSentiment) (2.27.1)\n","Requirement already satisfied:
urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (1.26.15)\n","Requirement already satisfied:
certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (2022.12.7)\n","Requirement already satisfied:
charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (2.0.12)\n","Requirement already satisfied:
idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (3.4)\n","Installing collected packages:
vaderSentiment\n","Successfully installed vaderSentiment-3.3.2\n"]}]},{"cell_type":"code","metadata":{"id":"dnS-5s6vZ3j8","executionInfo":
{"status":"ok","timestamp":1685957749070,"user_tz":-330,"elapsed":4,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}}},"source":["from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer\n","\n","#
SentimentIntensityAnalyzer.lexicon contains that\n","# dictionary of tokens and their scores that we talked about.\n","sa =
SentimentIntensityAnalyzer()"],"execution_count":41,"outputs":[]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"ghHt3ksJagxx","executionInfo":
{"status":"ok","timestamp":1685957749599,"user_tz":-330,"elapsed":532,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"0d1d0c83-509b-4922-80ff-fcd0703fd80a"},"source":["# A tokenizer should be good at
dealing with punctuation and emoticons (emojis) \n","# for VADER to work well. After all, emoticons are designed to convey a lot of \n","#
sentiment (emotion).\n","sa.lexicon"],"execution_count":42,"outputs":[{"output_type":"execute_result","data":{"text/plain":["{'$:': -1.5,\n"," '%)':
-0.4,\n"," '%-)': -1.5,\n"," '&-:': -0.4,\n"," '&:': -0.7,\n"," \"( '}{' )\": 1.6,\n"," '(%': -0.9,\n"," \"('-:\": 2.2,\n"," \"(':\": 2.3,\n"," '((-:': 2.1,\n"," '(*': 1.1,\n"," '(-%':
-0.7,\n"," '(-*': 1.3,\n"," '(-:': 1.6,\n"," '(-:0': 2.8,\n"," '(-:<': -0.4,\n"," '(-:o': 1.5,\n"," '(-:O': 1.5,\n"," '(-:{': -0.1,\n"," '(-:|>*': 1.9,\n"," '(-;': 1.3,\n"," '(-;|':
2.1,\n"," '(8': 2.6,\n"," '(:': 2.2,\n"," '(:0': 2.4,\n"," '(:<': -0.2,\n"," '(:o': 2.5,\n"," '(:O': 2.5,\n"," '(;': 1.1,\n"," '(;<': 0.3,\n"," '(=': 2.2,\n"," '(?:': 2.1,\n"," '(^:':
1.5,\n"," '(^;': 1.5,\n"," '(^;0': 2.0,\n"," '(^;o': 1.9,\n"," '(o:': 1.6,\n"," \")':\": -2.0,\n"," \")-':\": -2.1,\n"," ')-:': -2.1,\n"," ')-:<': -2.2,\n"," ')-:{': -2.1,\n"," '):':
-1.8,\n"," '):<': -1.9,\n"," '):{': -2.3,\n"," ');<': -2.6,\n"," '*)': 0.6,\n"," '*-)': 0.3,\n"," '*-:': 2.1,\n"," '*-;': 2.4,\n"," '*:': 1.9,\n"," '*<|:-)': 1.6,\n"," '*\\\\0/*':
2.3,\n"," '*^:': 1.6,\n"," ',-:': 1.2,\n"," \"---'-;-{@\": 2.3,\n"," '--<--<@': 2.2,\n"," '.-:': -1.2,\n"," '..###-:': -1.7,\n"," '..###:': -1.9,\n"," '/-:': -1.3,\n"," '/:':
```

```
-1.3,\n"," '/:<': -1.4,\n"," '/=': -0.9,\n"," '/^:': -1.0,\n"," '/o:': -1.4,\n"," '0-8': 0.1,\n"," '0-|': -1.2,\n"," '0:)': 1.9,\n"," '0:-)': 1.4,\n"," '0:-3': 1.5,\n"," '0:03':
1.9,\n"," '0;^)': 1.6,\n"," '0_o': -0.3,\n"," '10q': 2.1,\n"," '1337': 2.1,\n"," '143': 3.2,\n"," '1432': 2.6,\n"," '14aa41': 2.4,\n"," '182': -2.9,\n"," '187':
-3.1,\n"," '2g2b4g': 2.8,\n"," '2g2bt': -0.1,\n"," '2qt': 2.1,\n"," '3:(': -2.2,\n"," '3:)': 0.5,\n"," '3:-(': -2.3,\n"," '3:-)': -1.4,\n"," '4col': -2.2,\n"," '4q': -3.1,\n","
'5fs': 1.5,\n"," '8)': 1.9,\n"," '8-d': 1.7,\n"," '8-o': -0.3,\n"," '86': -1.6,\n"," '8d': 2.9,\n"," ':###..': -2.4,\n"," ':$': -0.2,\n"," ':&': -0.6,\n"," \":'(\": -2.2,\n","
\":')\": 2.3,\n"," \":'-(\": -2.4,\n"," \":'-)\": 2.7,\n"," ':(': -1.9,\n"," ':)': 2.0,\n"," ':*': 2.5,\n"," ':-###..': -2.5,\n"," ':-&': -0.5,\n"," ':-(': -1.5,\n"," ':-)': 1.3,\n"," ':-))':
2.8,\n"," ':-*': 1.7,\n"," ':-,': 1.1,\n"," ':-.': -0.9,\n"," ':-/': -1.2,\n"," ':-<': -1.5,\n"," ':-d': 2.3,\n"," ':-D': 2.3,\n"," ':-o': 0.1,\n"," ':-p': 1.5,\n"," ':-[': -1.6,\n"," ':-
\\\\': -0.9,\n"," ':-c': -1.3,\n"," ':-|': -0.7,\n"," ':-||': -2.5,\n"," ':-Þ': 0.9,\n"," ':/': -1.4,\n"," ':3': 2.3,\n"," ':<': -2.1,\n"," ':>': 2.1,\n"," ':?)': 1.3,\n"," ':?c':
-1.6,\n"," ':@': -2.5,\n"," ':d': 2.3,\n"," ':D': 2.3,\n"," ':l': -1.7,\n"," ':o': -0.4,\n"," ':p': 1.0,\n"," ':s': -1.2,\n"," ':[': -2.0,\n"," ':\\\\': -1.3,\n"," ':]': 2.2,\n"," ':^)':
2.1,\n"," ':^*': 2.6,\n"," ':^/': -1.2,\n"," ':^\\\\': -1.0,\n"," ':^|': -1.0,\n"," ':c': -2.1,\n"," ':c)': 2.0,\n"," ':o)': 2.1,\n"," ':o/': -1.4,\n"," ':o\\\\': -1.1,\n"," ':o|':
-0.6,\n"," ':P': 1.4,\n"," ':{': -1.9,\n"," ':|': -0.4,\n"," ':}': 2.1,\n"," ':Þ': 1.1,\n"," ';)': 0.9,\n"," ';-)': 1.0,\n"," ';-*': 2.2,\n"," ';-]': 0.7,\n"," ';d': 0.8,\n"," ';D':
0.8,\n"," ';]': 0.6,\n"," ';^)': 1.4,\n"," '-:': -2.0,\n"," '>.<': -1.3,\n"," '>:': -2.1,\n"," '>:(': -2.7,\n"," '>:)': 0.4,\n"," '>:-(': -2.7,\n"," '>:-)': -0.4,\n"," '>:/': -1.6,\n","
'>:o': -1.2,\n"," '>:p': 1.0,\n"," '>:[': -2.1,\n"," '>:\\\\': -1.7,\n"," '>;(': -2.9,\n"," '>;)': 0.1,\n"," '>_>^': 2.1,\n"," '@:': -2.1,\n"," '@>-->--': 2.1,\n"," \"@}-;-'---
\": 2.2,\n"," 'aas': 2.5,\n"," 'aayf': 2.7,\n"," 'afu': -2.9,\n"," 'alol': 2.8,\n"," 'ambw': 2.9,\n"," 'aml': 3.4,\n"," 'atab': -1.9,\n"," 'awol': -1.3,\n"," 'ayc': 0.2,\n","
'ayor': -1.2,\n"," 'aug-00': 0.3,\n"," 'bfd': -2.7,\n"," 'bfe': -2.6,\n"," 'bff': 2.9,\n"," 'bffn': 1.0,\n"," 'bl': 2.3,\n"," 'bsod': -2.2,\n"," 'btd': -2.1,\n"," 'btdt':
-0.1,\n"," 'bz': 0.4,\n"," 'b^d': 2.6,\n"," 'cwot': -2.3,\n"," \"d-':\": -2.5,\n"," 'd8': -3.2,\n"," 'd:': 1.2,\n"," 'd:<': -3.2,\n"," 'd;': -2.9,\n"," 'd=': 1.5,\n"," 'doa':
-2.3,\n"," 'dx': -3.0,\n"," 'ez': 1.5,\n"," 'fav': 2.0,\n"," 'fcol': -1.8,\n"," 'ff': 1.8,\n"," 'ffs': -2.8,\n"," 'fkm': -2.4,\n"," 'foaf': 1.8,\n"," 'ftw': 2.0,\n"," 'fu':
-3.7,\n"," 'fubar': -3.0,\n"," 'fwb': 2.5,\n"," 'fyi': 0.8,\n"," 'fysa': 0.4,\n"," 'g1': 1.4,\n"," 'gg': 1.2,\n"," 'gga': 1.7,\n"," 'gigo': -0.6,\n"," 'gj': 2.0,\n"," 'gl':
1.3,\n"," 'gla': 2.5,\n"," 'gn': 1.2,\n"," 'gr8': 2.7,\n"," 'grrr': -0.4,\n"," 'gt': 1.1,\n"," 'h&k': 2.3,\n"," 'hagd': 2.2,\n"," 'hagn': 2.2,\n"," 'hago': 1.2,\n"," 'hak':
1.9,\n"," 'hand': 2.2,\n"," 'heart': 3.2,\n"," 'hearts': 3.3,\n"," 'hho1/2k': 1.4,\n"," 'hhoj': 2.0,\n"," 'hhok': 0.9,\n"," 'hugz': 2.0,\n"," 'hi5': 1.9,\n"," 'idk':
-0.4,\n"," 'ijs': 0.7,\n"," 'ilu': 3.4,\n"," 'iluaaf': 2.7,\n"," 'ily': 3.4,\n"," 'ily2': 2.6,\n"," 'iou': 0.7,\n"," 'iyq': 2.3,\n"," 'j/j': 2.0,\n"," 'j/k': 1.6,\n"," 'j/p': 1.4,\n","
'j/t': -0.2,\n"," 'j/w': 1.0,\n"," 'j4f': 1.4,\n"," 'j4g': 1.7,\n"," 'jho': 0.8,\n"," 'jhomf': 1.0,\n"," 'jj': 1.0,\n"," 'jk': 0.9,\n"," 'jp': 0.8,\n"," 'jt': 0.9,\n"," 'jw': 1.6,\n","
'jealz': -1.2,\n"," 'k4y': 2.3,\n"," 'kfy': 2.3,\n"," 'kia': -3.2,\n"," 'kk': 1.5,\n"," 'kmuf': 2.2,\n"," 'l': 2.0,\n"," 'l&r': 2.2,\n"," 'laoj': 1.3,\n"," 'lmao': 2.9,\n","
'lmbao': 1.8,\n"," 'lmfao': 2.5,\n"," 'lmso': 2.7,\n"," 'lol': 1.8,\n"," 'lolz': 2.7,\n"," 'lts': 1.6,\n"," 'ly': 2.6,\n"," 'ly4e': 2.7,\n"," 'lya': 3.3,\n"," 'lyb': 3.0,\n","
'lyl': 3.1,\n"," 'lylab': 2.7,\n"," 'lylas': 2.6,\n"," 'lylb': 1.6,\n"," 'm8': 1.4,\n"," 'mia': -1.2,\n"," 'mml': 2.0,\n"," 'mofo': -2.4,\n"," 'muah': 2.3,\n"," 'mubar':
-1.0,\n"," 'musm': 0.9,\n"," 'mwah': 2.5,\n"," 'n1': 1.9,\n"," 'nbd': 1.3,\n"," 'nbif': -0.5,\n"," 'nfc': -2.7,\n"," 'nfw': -2.4,\n"," 'nh': 2.2,\n"," 'nimby': -0.8,\n","
'nimjd': -0.7,\n"," 'nimq': -0.2,\n"," 'nimy': -1.4,\n"," 'nitl': -1.5,\n"," 'nme': -2.1,\n"," 'noyb': -0.7,\n"," 'np': 1.4,\n"," 'ntmu': 1.4,\n"," 'o-8': -0.5,\n"," 'o-:':
-0.3,\n"," 'o-|': -1.1,\n"," 'o.o': -0.8,\n"," 'O.o': -0.6,\n"," 'o.O': -0.6,\n"," 'o:': -0.2,\n"," 'o:)': 1.5,\n"," 'o:-)': 2.0,\n"," 'o:-3': 2.2,\n"," 'o:3': 2.3,\n"," 'o:<':
-0.3,\n"," 'o;^)': 1.6,\n"," 'ok': 1.2,\n"," 'o_o': -0.5,\n"," 'O_o': -0.5,\n"," 'o_O': -0.5,\n"," 'pita': -2.4,\n"," 'pls': 0.3,\n"," 'plz': 0.3,\n"," 'pmbi': 0.8,\n","
'pmfji': 0.3,\n"," 'pmji': 0.7,\n"," 'po': -2.6,\n"," 'ptl': 2.6,\n"," 'pu': -1.1,\n"," 'qq': -2.2,\n"," 'qt': 1.8,\n"," 'r&r': 2.4,\n"," 'rofl': 2.7,\n"," 'roflmao': 2.5,\n","
'rotfl': 2.6,\n"," 'rotflmao': 2.8,\n"," 'rotflmfao': 2.5,\n"," 'rotflol': 3.0,\n"," 'rotgl': 2.9,\n"," 'rotglmao': 1.8,\n"," 's:': -1.1,\n"," 'sapfu': -1.1,\n"," 'sete':
2.8,\n"," 'sfete': 2.7,\n"," 'sgtm': 2.4,\n"," 'slap': 0.6,\n"," 'slaw': 2.1,\n"," 'smh': -1.3,\n"," 'snafu': -2.5,\n"," 'sob': -1.0,\n"," 'swak': 2.3,\n"," 'tgif':
2.3,\n"," 'thks': 1.4,\n"," 'thx': 1.5,\n"," 'tia': 2.3,\n"," 'tmi': -0.3,\n"," 'tnx': 1.1,\n"," 'true': 1.8,\n"," 'tx': 1.5,\n"," 'txs': 1.1,\n"," 'ty': 1.6,\n"," 'tyvm':
2.5,\n"," 'urw': 1.9,\n"," 'vbg': 2.1,\n"," 'vbs': 3.1,\n"," 'vip': 2.3,\n"," 'vwd': 2.6,\n"," 'vwp': 2.1,\n"," 'wag': -0.2,\n"," 'wd': 2.7,\n"," 'wilco': 0.9,\n"," 'wp':
1.0,\n"," 'wtf': -2.8,\n"," 'wtg': 2.1,\n"," 'wth': -2.4,\n"," 'x-d': 2.6,\n"," 'x-p': 1.7,\n"," 'xd': 2.8,\n"," 'xlnt': 3.0,\n"," 'xoxo': 3.0,\n"," 'xoxozzz': 2.3,\n"," 'xp':
1.6,\n"," 'xqzt': 1.6,\n"," 'xtc': 0.8,\n"," 'yolo': 1.1,\n"," 'yoyo': 0.4,\n"," 'yvw': 1.6,\n"," 'yw': 1.8,\n"," 'ywia': 2.5,\n"," 'zzz': -1.2,\n"," '[-;': 0.5,\n"," '[:':
1.3,\n"," '[;': 1.0,\n"," '[=': 1.7,\n"," '\\\\-:': -1.0,\n"," '\\\\:': -1.0,\n"," '\\\\:<': -1.7,\n"," '\\\\=': -1.1,\n"," '\\\\^:': -1.3,\n"," '\\\\o/': 2.2,\n"," '\\\\o:': -1.2,\n"," ']-:':
-2.1,\n"," ']:': -1.6,\n"," ']:<': -2.5,\n"," '^<_<': 1.4,\n"," '^urs': -2.8,\n"," 'abandon': -1.9,\n"," 'abandoned': -2.0,\n"," 'abandoner': -1.9,\n","
'abandoners': -1.9,\n"," 'abandoning': -1.6,\n"," 'abandonment': -2.4,\n"," 'abandonments': -1.7,\n"," 'abandons': -1.3,\n"," 'abducted': -2.3,\n","
'abduction': -2.8,\n"," 'abductions': -2.0,\n"," 'abhor': -2.0,\n"," 'abhorred': -2.4,\n"," 'abhorrent': -3.1,\n"," 'abhors': -2.9,\n"," 'abilities': 1.0,\n","
'ability': 1.3,\n"," 'aboard': 0.1,\n"," 'absentee': -1.1,\n"," 'absentees': -0.8,\n"," 'absolve': 1.2,\n"," 'absolved': 1.5,\n"," 'absolves': 1.3,\n"," 'absolving':
1.6,\n"," 'abuse': -3.2,\n"," 'abused': -2.3,\n"," 'abuser': -2.6,\n"," 'abusers': -2.6,\n"," 'abuses': -2.6,\n"," 'abusing': -2.0,\n"," 'abusive': -3.2,\n","
'abusively': -2.8,\n"," 'abusiveness': -2.5,\n"," 'abusivenesses': -3.0,\n"," 'accept': 1.6,\n"," 'acceptabilities': 1.6,\n"," 'acceptability': 1.1,\n","
'acceptable': 1.3,\n"," 'acceptableness': 1.3,\n"," 'acceptably': 1.5,\n"," 'acceptance': 2.0,\n"," 'acceptances': 1.7,\n"," 'acceptant': 1.6,\n","
'acceptation': 1.3,\n"," 'acceptations': 0.9,\n"," 'accepted': 1.1,\n"," 'accepting': 1.6,\n"," 'accepts': 1.3,\n"," 'accident': -2.1,\n"," 'accidental': -0.3,\n","
'accidentally': -1.4,\n"," 'accidents': -1.3,\n"," 'accomplish': 1.8,\n"," 'accomplished': 1.9,\n"," 'accomplishes': 1.7,\n"," 'accusation': -1.0,\n","
'accusations': -1.3,\n"," 'accuse': -0.8,\n"," 'accused': -1.2,\n"," 'accuses': -1.4,\n"," 'accusing': -0.7,\n"," 'ache': -1.6,\n"," 'ached': -1.6,\n"," 'aches':
-1.0,\n"," 'achievable': 1.3,\n"," 'aching': -2.2,\n"," 'acquit': 0.8,\n"," 'acquits': 0.1,\n"," 'acquitted': 1.0,\n"," 'acquitting': 1.3,\n"," 'acrimonious':
-1.7,\n"," 'active': 1.7,\n"," 'actively': 1.3,\n"," 'activeness': 0.6,\n"," 'activenesses': 0.8,\n"," 'actives': 1.1,\n"," 'adequate': 0.9,\n"," 'admirability':
2.4,\n"," 'admirable': 2.6,\n"," 'admirableness': 2.2,\n"," 'admirably': 2.5,\n"," 'admiral': 1.3,\n"," 'admirals': 1.5,\n"," 'admiralties': 1.6,\n"," 'admiralty':
1.2,\n"," 'admiration': 2.5,\n"," 'admirations': 1.6,\n"," 'admire': 2.1,\n"," 'admired': 2.3,\n"," 'admirer': 1.8,\n"," 'admirers': 1.7,\n"," 'admires': 1.5,\n","
'admiring': 1.6,\n"," 'admiringly': 2.3,\n"," 'admit': 0.8,\n"," 'admits': 1.2,\n"," 'admitted': 0.4,\n"," 'admonished': -1.9,\n"," 'adopt': 0.7,\n"," 'adopts':
0.7,\n"," 'adorability': 2.2,\n"," 'adorable': 2.2,\n"," 'adorableness': 2.5,\n"," 'adorably': 2.1,\n"," 'adoration': 2.9,\n"," 'adorations': 2.2,\n"," 'adore':
2.6,\n"," 'adored': 1.8,\n"," 'adorer': 1.7,\n"," 'adorers': 2.1,\n"," 'adores': 1.6,\n"," 'adoring': 2.6,\n"," 'adoringly': 2.4,\n"," 'adorn': 0.9,\n"," 'adorned':
0.8,\n"," 'adorner': 1.3,\n"," 'adorners': 0.9,\n"," 'adorning': 1.0,\n"," 'adornment': 1.3,\n"," 'adornments': 0.8,\n"," 'adorns': 0.5,\n"," 'advanced':
1.0,\n"," 'advantage': 1.0,\n"," 'advantaged': 1.4,\n"," 'advantageous': 1.5,\n"," 'advantageously': 1.9,\n"," 'advantageousness': 1.6,\n","
'advantages': 1.5,\n"," 'advantaging': 1.6,\n"," 'adventure': 1.3,\n"," 'adventured': 1.3,\n"," 'adventurer': 1.2,\n"," 'adventurers': 0.9,\n"," 'adventures':
1.4,\n"," 'adventuresome': 1.7,\n"," 'adventuresomeness': 1.3,\n"," 'adventuress': 0.8,\n"," 'adventuresses': 1.4,\n"," 'adventuring': 2.3,\n","
'adventurism': 1.5,\n"," 'adventurist': 1.4,\n"," 'adventuristic': 1.7,\n"," 'adventurists': 1.2,\n"," 'adventurous': 1.4,\n"," 'adventurously': 1.3,\n","
'adventurousness': 1.8,\n"," 'adversarial': -1.5,\n"," 'adversaries': -1.0,\n"," 'adversary': -0.8,\n"," 'adversative': -1.2,\n"," 'adversatively': -0.1,\n","
'adversatives': -1.0,\n"," 'adverse': -1.5,\n"," 'adversely': -0.8,\n"," 'adverseness': -0.6,\n"," 'adversities': -1.5,\n"," 'adversity': -1.8,\n"," 'affected':
-0.6,\n"," 'affection': 2.4,\n"," 'affectional': 1.9,\n"," 'affectionally': 1.5,\n"," 'affectionate': 1.9,\n"," 'affectionately': 2.2,\n"," 'affectioned': 1.8,\n","
'affectionless': -2.0,\n"," 'affections': 1.5,\n"," 'afflicted': -1.5,\n"," 'affronted': 0.2,\n"," 'aggravate': -2.5,\n"," 'aggravated': -1.9,\n"," 'aggravates':
-1.9,\n"," 'aggravating': -1.2,\n"," 'aggress': -1.3,\n"," 'aggressed': -1.4,\n"," 'aggresses': -0.5,\n"," 'aggressing': -0.6,\n"," 'aggression': -1.2,\n","
'aggressions': -1.3,\n"," 'aggressive': -0.6,\n"," 'aggressively': -1.3,\n"," 'aggressiveness': -1.8,\n"," 'aggressivities': -1.4,\n"," 'aggressivity': -0.6,\n","
'aggressor': -0.8,\n"," 'aggressors': -0.9,\n"," 'aghast': -1.9,\n"," 'agitate': -1.7,\n"," 'agitated': -2.0,\n"," 'agitatedly': -1.6,\n"," 'agitates': -1.4,\n","
'agitating': -1.8,\n"," 'agitation': -1.0,\n"," 'agitational': -1.2,\n"," 'agitations': -1.3,\n"," 'agitative': -1.3,\n"," 'agitato': -0.1,\n"," 'agitator': -1.4,\n","
'agitators': -2.1,\n"," 'agog': 1.9,\n"," 'agonise': -2.1,\n"," 'agonised': -2.3,\n"," 'agonises': -2.4,\n"," 'agonising': -1.5,\n"," 'agonize': -2.3,\n","
'agonized': -2.2,\n"," 'agonizes': -2.3,\n"," 'agonizing': -2.7,\n"," 'agonizingly': -2.3,\n"," 'agony': -1.8,\n"," 'agree': 1.5,\n"," 'agreeability': 1.9,\n","
'agreeable': 1.8,\n"," 'agreeableness': 1.8,\n"," 'agreeablenesses': 1.3,\n"," 'agreeably': 1.6,\n"," 'agreed': 1.1,\n"," 'agreeing': 1.4,\n"," 'agreement':
2.2,\n"," 'agreements': 1.1,\n"," 'agrees': 0.8,\n"," 'alarm': -1.4,\n"," 'alarmed': -1.4,\n"," 'alarming': -0.5,\n"," 'alarmingly': -2.6,\n"," 'alarmism':
-0.3,\n"," 'alarmists': -1.1,\n"," 'alarms': -1.1,\n"," 'alas': -1.1,\n"," 'alert': 1.2,\n"," 'alienation': -1.1,\n"," 'alive': 1.6,\n"," 'allergic': -1.2,\n"," 'allow':
0.9,\n"," 'alone': -1.0,\n"," 'alright': 1.0,\n"," 'amaze': 2.5,\n"," 'amazed': 2.2,\n"," 'amazedly': 2.1,\n"," 'amazement': 2.5,\n"," 'amazements': 2.2,\n","
'amazes': 2.2,\n"," 'amazing': 2.8,\n"," 'amazon': 0.7,\n"," 'amazonite': 0.2,\n"," 'amazons': -0.1,\n"," 'amazonstone': 1.0,\n"," 'amazonstones':
0.2,\n"," 'ambitious': 2.1,\n"," 'ambivalent': 0.5,\n"," 'amor': 3.0,\n"," 'amoral': -1.6,\n"," 'amoralism': -0.7,\n"," 'amoralisms': -0.7,\n"," 'amoralities':
-1.2,\n"," 'amorality': -1.5,\n"," 'amorally': -1.0,\n"," 'amoretti': 0.2,\n"," 'amoretto': 0.6,\n"," 'amorettos': 0.3,\n"," 'amorino': 1.2,\n"," 'amorist': 1.6,\n","
'amoristic': 1.0,\n"," 'amorists': 0.1,\n"," 'amoroso': 2.3,\n"," 'amorous': 1.8,\n"," 'amorously': 2.3,\n"," 'amorousness': 2.0,\n"," 'amorphous': -0.2,\n","
'amorphously': 0.1,\n"," 'amorphousness': 0.3,\n"," 'amort': -2.1,\n"," 'amortise': 0.5,\n"," 'amortised': -0.2,\n"," 'amortises': 0.1,\n"," 'amortizable':
0.5,\n"," 'amortization': 0.6,\n"," 'amortizations': 0.2,\n"," 'amortize': -0.1,\n"," 'amortized': 0.8,\n"," 'amortizes': 0.6,\n"," 'amortizing': 0.8,\n","
'amusable': 0.7,\n"," 'amuse': 1.7,\n"," 'amused': 1.8,\n"," 'amusedly': 2.2,\n"," 'amusement': 1.5,\n"," 'amusements': 1.5,\n"," 'amuser': 1.1,\n","
'amusers': 1.3,\n"," 'amuses': 1.7,\n"," 'amusia': 0.3,\n"," 'amusias': -0.4,\n"," 'amusing': 1.6,\n"," 'amusingly': 0.8,\n"," 'amusingness': 1.8,\n","
'amusive': 1.7,\n"," 'anger': -2.7,\n"," 'angered': -2.3,\n"," 'angering': -2.2,\n"," 'angerly': -1.9,\n"," 'angers': -2.3,\n"," 'angrier': -2.3,\n"," 'angriest':
-3.1,\n"," 'angrily': -1.8,\n"," 'angriness': -1.7,\n"," 'angry': -2.3,\n"," 'anguish': -2.9,\n"," 'anguished': -1.8,\n"," 'anguishes': -2.1,\n"," 'anguishing':
-2.7,\n"," 'animosity': -1.9,\n"," 'annoy': -1.9,\n"," 'annoyance': -1.3,\n"," 'annoyances': -1.8,\n"," 'annoyed': -1.6,\n"," 'annoyer': -2.2,\n"," 'annoyers':
-1.5,\n"," 'annoying': -1.7,\n"," 'annoys': -1.8,\n"," 'antagonism': -1.9,\n"," 'antagonisms': -1.2,\n"," 'antagonist': -1.9,\n"," 'antagonistic': -1.7,\n","
'antagonistically': -2.2,\n"," 'antagonists': -1.7,\n"," 'antagonize': -2.0,\n"," 'antagonized': -1.4,\n"," 'antagonizes': -0.5,\n"," 'antagonizing': -2.7,\n","
```

'anti': -1.3,\n"," 'anticipation': 0.4,\n"," 'anxieties': -0.6,\n"," 'anxiety': -0.7,\n"," 'anxious': -1.0,\n"," 'anxiously': -0.9,\n"," 'anxiousness': -1.0,\n"," 'aok':
2.0,\n"," 'apathetic': -1.2,\n"," 'apathetically': -0.4,\n"," 'apathies': -0.6,\n"," 'apathy': -1.2,\n"," 'apeshit': -0.9,\n"," 'apocalyptic': -3.4,\n"," 'apologise':
1.6,\n"," 'apologised': 0.4,\n"," 'apologises': 0.8,\n"," 'apologising': 0.2,\n"," 'apologize': 0.4,\n"," 'apologized': 1.3,\n"," 'apologizes': 1.5,\n","
'apologizing': -0.3,\n"," 'apology': 0.2,\n"," 'appall': -2.4,\n"," 'appalled': -2.0,\n"," 'appalling': -1.5,\n"," 'appallingly': -2.0,\n"," 'appalls': -1.9,\n","
'appease': 1.1,\n"," 'appeased': 0.9,\n"," 'appeases': 0.9,\n"," 'appeasing': 1.0,\n"," 'applaud': 2.0,\n"," 'applauded': 1.5,\n"," 'applauding': 2.1,\n","
'applauds': 1.4,\n"," 'applause': 1.8,\n"," 'appreciate': 1.7,\n"," 'appreciated': 2.3,\n"," 'appreciates': 2.3,\n"," 'appreciating': 1.9,\n"," 'appreciation':
2.3,\n"," 'appreciations': 1.7,\n"," 'appreciative': 2.6,\n"," 'appreciatively': 1.8,\n"," 'appreciativeness': 1.6,\n"," 'appreciator': 2.6,\n"," 'appreciators':
1.5,\n"," 'appreciatory': 1.7,\n"," 'apprehensible': 1.1,\n"," 'apprehensibly': -0.2,\n"," 'apprehension': -2.1,\n"," 'apprehensions': -0.9,\n","
'apprehensively': -0.3,\n"," 'apprehensiveness': -0.7,\n"," 'approval': 2.1,\n"," 'approved': 1.8,\n"," 'approves': 1.7,\n"," 'ardent': 2.1,\n"," 'arguable':
-1.0,\n"," 'arguably': -1.0,\n"," 'argue': -1.4,\n"," 'argued': -1.5,\n"," 'arguer': -1.6,\n"," 'arguers': -1.4,\n"," 'argues': -1.6,\n"," 'arguing': -2.0,\n","
'argument': -1.5,\n"," 'argumentative': -1.5,\n"," 'argumentatively': -1.8,\n"," 'argumentive': -1.5,\n"," 'arguments': -1.7,\n"," 'arrest': -1.4,\n","
'arrested': -2.1,\n"," 'arrests': -1.9,\n"," 'arrogance': -2.4,\n"," 'arrogances': -1.9,\n"," 'arrogant': -2.2,\n"," 'arrogantly': -1.8,\n"," 'ashamed': -2.1,\n","
'ashamedly': -1.7,\n"," 'ass': -2.5,\n"," 'assassination': -2.9,\n"," 'assassinations': -2.7,\n"," 'assault': -2.8,\n"," 'assaulted': -2.4,\n"," 'assaulting':
-2.3,\n"," 'assaultive': -2.8,\n"," 'assaults': -2.5,\n"," 'asset': 1.5,\n"," 'assets': 0.7,\n"," 'assfucking': -2.5,\n"," 'assholes': -2.8,\n"," 'assurance':
1.4,\n"," 'assurances': 1.4,\n"," 'assure': 1.4,\n"," 'assured': 1.5,\n"," 'assuredly': 1.6,\n"," 'assuredness': 1.4,\n"," 'assurer': 0.9,\n"," 'assurers':
1.1,\n"," 'assures': 1.3,\n"," 'assurgent': 1.3,\n"," 'assuring': 1.6,\n"," 'assuror': 0.5,\n"," 'assurors': 0.7,\n"," 'astonished': 1.6,\n"," 'astound': 1.7,\n","
'astounded': 1.8,\n"," 'astounding': 1.8,\n"," 'astoundingly': 2.1,\n"," 'astounds': 2.1,\n"," 'attachment': 1.2,\n"," 'attachments': 1.1,\n"," 'attack':
-2.1,\n"," 'attacked': -2.0,\n"," 'attacker': -2.7,\n"," 'attackers': -2.7,\n"," 'attacking': -2.0,\n"," 'attacks': -1.9,\n"," 'attract': 1.5,\n"," 'attractancy':
0.9,\n"," 'attractant': 1.3,\n"," 'attractants': 1.4,\n"," 'attracted': 1.8,\n"," 'attracting': 2.1,\n"," 'attraction': 2.0,\n"," 'attractions': 1.8,\n"," 'attractive':
1.9,\n"," 'attractively': 2.2,\n"," 'attractiveness': 1.8,\n"," 'attractivenesses': 2.1,\n"," 'attractor': 1.2,\n"," 'attractors': 1.2,\n"," 'attracts': 1.7,\n","
'audacious': 0.9,\n"," 'authority': 0.3,\n"," 'aversion': -1.9,\n"," 'aversions': -1.1,\n"," 'aversive': -1.6,\n"," 'aversively': -0.8,\n"," 'avert': -0.7,\n","
'averted': -0.3,\n"," 'averts': -0.4,\n"," 'avid': 1.2,\n"," 'avoid': -1.2,\n"," 'avoidance': -1.7,\n"," 'avoidances': -1.1,\n"," 'avoided': -1.4,\n"," 'avoider':
-1.8,\n"," 'avoiders': -1.4,\n"," 'avoiding': -1.4,\n"," 'avoids': -0.7,\n"," 'await': 0.4,\n"," 'awaited': -0.1,\n"," 'awaits': 0.3,\n"," 'award': 2.5,\n","
'awardable': 2.4,\n"," 'awarded': 1.7,\n"," 'awardee': 1.8,\n"," 'awardees': 1.2,\n"," 'awarder': 0.9,\n"," 'awarders': 1.3,\n"," 'awarding': 1.9,\n","
'awards': 2.0,\n"," 'awesome': 3.1,\n"," 'awful': -2.0,\n"," 'awkward': -0.6,\n"," 'awkwardly': -1.3,\n"," 'awkwardness': -0.7,\n"," 'axe': -0.4,\n"," 'axed':
-1.3,\n"," 'backed': 0.1,\n"," 'backing': 0.1,\n"," 'backs': -0.2,\n"," 'bad': -2.5,\n"," 'badass': 1.4,\n"," 'badly': -2.1,\n"," 'bailout': -0.4,\n"," 'bamboozle':
-1.5,\n"," 'bamboozled': -1.5,\n"," 'bamboozles': -1.5,\n"," 'ban': -2.6,\n"," 'banish': -1.9,\n"," 'bankrupt': -2.6,\n"," 'bankster': -2.1,\n"," 'banned':
-2.0,\n"," 'bargain': 0.8,\n"," 'barrier': -0.5,\n"," 'bashful': -0.1,\n"," 'bashfully': 0.2,\n"," 'bashfulness': -0.8,\n"," 'bastard': -2.5,\n"," 'bastardies':
-1.8,\n"," 'bastardise': -2.1,\n"," 'bastardised': -2.3,\n"," 'bastardises': -2.3,\n"," 'bastardising': -2.6,\n"," 'bastardization': -2.4,\n"," 'bastardizations':
-2.1,\n"," 'bastardize': -2.4,\n"," 'bastardized': -2.0,\n"," 'bastardizes': -1.8,\n"," 'bastardizing': -2.3,\n"," 'bastardly': -2.7,\n"," 'bastards': -3.0,\n","
'bastardy': -2.7,\n"," 'battle': -1.6,\n"," 'battled': -1.2,\n"," 'battlefield': -1.6,\n"," 'battlefields': -0.9,\n"," 'battlefront': -1.2,\n"," 'battlefronts': -0.8,\n","
'battleground': -1.7,\n"," 'battlegrounds': -0.6,\n"," 'battlement': -0.4,\n"," 'battlements': -0.4,\n"," 'battler': -0.8,\n"," 'battlers': -0.2,\n"," 'battles':
-1.6,\n"," 'battleship': -0.1,\n"," 'battleships': -0.5,\n"," 'battlewagon': -0.3,\n"," 'battlewagons': -0.5,\n"," 'battling': -1.1,\n"," 'beaten': -1.8,\n","
'beatific': 1.8,\n"," 'beating': -2.0,\n"," 'beaut': 1.6,\n"," 'beauteous': 2.5,\n"," 'beauteously': 2.6,\n"," ...}"]},"metadata":{},"execution_count":42}]},
{"cell_type":"markdown","metadata":{"id":"k30xatt3cniB"},"source":["{ ... \\\\\n","':(': -1.9, \\\\\n","':)': 2.0, \\\\\n","... \\\\\n","'pls': 0.3, \\\\\n","'plz': 0.3,
\\\\\n","... \\\\\n","'great ': 3.1, \\\\\n","... } \\\\\n","\n","\n","If you use a stemmer (or lemmatizer) in your pipeline, you'll need to\n","apply that stemmer
to the VADER lexicon, too, combining the scores\n","for all the words that go together in a single stem or lemma."]},
{"cell_type":"code","metadata":{"colab":{"base_uri":"https://localhost:8080/"},"id":"cbfFcSHna9WQ","executionInfo":
{"status":"ok","timestamp":1685957749599,"user_tz":-330,"elapsed":24,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"b549d0cc-0ecf-4d90-d091-0f3a1b9d03c7"},"source":["[(tok, score) for tok, score in
sa.lexicon.items() if \" \" in tok]"],"execution_count":43,"outputs":[{"output_type":"execute_result","data":{"text/plain":["[(\"('}{')\", 1.6),\n"," (\"can't
stand\", -2.0),\n"," ('fed up', -1.8),\n"," ('screwed up', -1.5)]"]},"metadata":{},"execution_count":43}]},{"cell_type":"markdown","metadata":
{"id":"qq0B9a4xbFZR"},"source":["Out of 7500 tokens defined in VADER, only 4\n","contain spaces, and only 3 of those are actually\n","n-grams;
the other is an emoticon for “kiss.”"]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"pllYsw3edeCY","executionInfo":
{"status":"ok","timestamp":1685957749599,"user_tz":-330,"elapsed":9,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"52c58390-cf4f-40ef-f014-408d20c6100b"},"source":["sa.polarity_scores(text=\"Python is
very readable and it's great for NLP.\")"],"execution_count":44,"outputs":[{"output_type":"execute_result","data":{"text/plain":["{'neg': 0.0, 'neu':
0.661, 'pos': 0.339, 'compound': 0.6249}"]},"metadata":{},"execution_count":44}]},{"cell_type":"markdown","metadata":
{"id":"8Okn4U5dd2Yz"},"source":["The VADER algorithm considers the intensity of\n","sentiment polarity in three separate scores
(positive,\n","negative, and neutral) and then combines them\n","together into a compound positivity sentiment."]},{"cell_type":"code","metadata":
{"colab":{"base_uri":"https://localhost:8080/"},"id":"nkug0oZQdfiW","executionInfo":
{"status":"ok","timestamp":1685957749599,"user_tz":-330,"elapsed":7,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"3674a40f-0f71-4d7f-b8d0-fa0c9f6bb7f0"},"source":["sa.polarity_scores(text=\"Python is not
a bad choice for most applications.\")"],"execution_count":45,"outputs":[{"output_type":"execute_result","data":{"text/plain":["{'neg': 0.0, 'neu':
0.737, 'pos': 0.263, 'compound': 0.431}"]},"metadata":{},"execution_count":45}]},{"cell_type":"markdown","metadata":
{"id":"rNIrSW9zduUn"},"source":["Notice that VADER handles negation pretty well—“great”\n","has a slightly more positive sentiment than “not
bad.”\n","VADER’s built-in tokenizer ignores any words that aren’t\n","in its lexicon, and it doesn’t consider n-grams at all."]},
{"cell_type":"markdown","metadata":{"id":"EIcPnq52d9m7"},"source":["Let’s see how well this rule-based approach does for the example
statements we mentioned earlier:"]},{"cell_type":"code","metadata":{"id":"dkvEiT_3eATO","executionInfo":
{"status":"ok","timestamp":1685957749600,"user_tz":-330,"elapsed":7,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}}},"source":["corpus = [\"Absolutely perfect! Love it! :-) :-) :-)\",\n"," \"Horrible! Completely useless. :
(\",\n"," \"It was OK. Some good and some bad things.\"]"],"execution_count":46,"outputs":[]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"ZAOoBmLHeED-","executionInfo":
{"status":"ok","timestamp":1685957749600,"user_tz":-330,"elapsed":7,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"b4b42866-f771-45f8-8c6b-df5afbc86818"},"source":["for doc in corpus:\n"," scores =
sa.polarity_scores(doc)\n"," print('{:+}: {}'.format(scores['compound'], doc))"],"execution_count":47,"outputs":
[{"output_type":"stream","name":"stdout","text":["+0.9428: Absolutely perfect! Love it! :-) :-) :-)\n","-0.8768: Horrible! Completely useless. :
(\n","-0.1531: It was OK. Some good and some bad things.\n"]}]},{"cell_type":"markdown","metadata":{"id":"7XlHPzU7eIXH"},"source":["The only
drawback is that VADER doesn’t look at all the words in a document, only about 7,500. \n","\n","What if you want all the words to help add to the
sentiment score? \n","\n","And what if you don’t want to have to code your own understanding of the words in a dictionary of thousands of words
or add a bunch of custom words to the dictionary?\n","\n","That’s what machine learning sentiment analyzers are for."]},
{"cell_type":"markdown","source":["## **Naive Bayes**\n","\n","A Naive Bayes model tries to find keywords in a set of documents that are
predictive of your target (output) variable. \n","\n","When your target variable is the sentiment you are trying to predict, the model will find words
that predict that sentiment. \n","\n","The nice thing about a Naive Bayes model is that the internal coefficients will map words or tokens to scores
just like VADER does. Only this time you won’t have to be limited to just what\n","an individual human decided those scores should be. The
machine will find the “best” scores for any problem.\n"],"metadata":{"id":"ag0s11FBeQU2"}},{"cell_type":"markdown","source":["For any machine
learning algorithm, you first need to find a dataset. \n","\n","You need a bunch of text documents that have labels for their positive emotional
content (positivity sentiment). \n","\n","Hutto compiled four different sentiment datasets for us when he and his collaborators built VADER. You’ll
load them from the `nlpia` package:"],"metadata":{"id":"2S7ydm-2eRIy"}},{"cell_type":"code","source":["pip install nlpia"],"metadata":
{"id":"a1Poab_yfzm7","colab":{"base_uri":"https://localhost:8080/"},"executionInfo":
{"status":"ok","timestamp":1685957778979,"user_tz":-330,"elapsed":29384,"user":{"displayName":"Dr. Muneendra
Ojha","userId":"06154531826228794240"}},"outputId":"117ca697-d642-4c89-cef0-ed8fd6c10b4f"},"execution_count":48,"outputs":
[{"output_type":"stream","name":"stdout","text":["Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/\n","Collecting nlpia\n"," Downloading nlpia-0.5.2-py2.py3-none-any.whl (32.0 MB)\n","\u001b[2K

```
\u001b[90m━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━\u001b[0m \u001b[32m32.0/32.0
MB\u001b[0m \u001b[31m10.8 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hRequirement already satisfied: future in
/usr/local/lib/python3.10/dist-packages (from nlpia) (0.18.3)\n","Collecting jupyter (from nlpia)\n"," Downloading jupyter-1.0.0-py2.py3-none-
any.whl (2.7 kB)\n","Requirement already satisfied: h5py in /usr/local/lib/python3.10/dist-packages (from nlpia) (3.8.0)\n","Requirement already
satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from nlpia) (3.7.1)\n","Requirement already satisfied: nltk in
/usr/local/lib/python3.10/dist-packages (from nlpia) (3.8.1)\n","Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(from nlpia) (1.5.3)\n","Collecting pypandoc (from nlpia)\n"," Downloading pypandoc-1.11-py3-none-any.whl (20 kB)\n","Requirement already
satisfied: plotly in /usr/local/lib/python3.10/dist-packages (from nlpia) (5.13.1)\n","Collecting python-Levenshtein (from nlpia)\n"," Downloading
python_Levenshtein-0.21.0-py3-none-any.whl (9.4 kB)\n","Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages
(from nlpia) (1.2.2)\n","Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (from nlpia) (0.12.2)\n","Requirement
already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nlpia) (4.65.0)\n","Requirement already satisfied: gensim in
/usr/local/lib/python3.10/dist-packages (from nlpia) (4.3.1)\n","Requirement already satisfied: pandas-datareader in /usr/local/lib/python3.10/dist-
packages (from nlpia) (0.10.0)\n","Collecting pugnlp (from nlpia)\n"," Downloading pugnlp-0.2.6-py2.py3-none-any.whl (706 kB)\n","\u001b[2K
\u001b[90m━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━\u001b[0m \u001b[32m706.6/706.6
kB\u001b[0m \u001b[31m39.5 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hRequirement already satisfied: tensorflow in
/usr/local/lib/python3.10/dist-packages (from nlpia) (2.12.0)\n","Requirement already satisfied: keras in /usr/local/lib/python3.10/dist-packages
(from nlpia) (2.12.0)\n","Requirement already satisfied: regex in /usr/local/lib/python3.10/dist-packages (from nlpia) (2022.10.31)\n","Requirement
already satisfied: spacy in /usr/local/lib/python3.10/dist-packages (from nlpia) (3.5.2)\n","Requirement already satisfied: lxml in
/usr/local/lib/python3.10/dist-packages (from nlpia) (4.9.2)\n","Collecting html2text (from nlpia)\n"," Downloading html2text-2020.1.16-py3-none-
any.whl (32 kB)\n","Requirement already satisfied: html5lib in /usr/local/lib/python3.10/dist-packages (from nlpia) (1.1)\n","Requirement already
satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from gensim->nlpia) (1.22.4)\n","Requirement already satisfied: scipy>=1.7.0
in /usr/local/lib/python3.10/dist-packages (from gensim->nlpia) (1.10.1)\n","Requirement already satisfied: smart-open>=1.8.1 in
/usr/local/lib/python3.10/dist-packages (from gensim->nlpia) (6.3.0)\n","Requirement already satisfied: six>=1.9 in /usr/local/lib/python3.10/dist-
packages (from html5lib->nlpia) (1.16.0)\n","Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from
html5lib->nlpia) (0.5.1)\n","Requirement already satisfied: notebook in /usr/local/lib/python3.10/dist-packages (from jupyter->nlpia)
(6.4.8)\n","Collecting qtconsole (from jupyter->nlpia)\n"," Downloading qtconsole-5.4.3-py3-none-any.whl (121 kB)\n","\u001b[2K
\u001b[90m━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━\u001b[0m \u001b[32m121.9/121.9
kB\u001b[0m \u001b[31m14.6 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hRequirement already satisfied: jupyter-console
in /usr/local/lib/python3.10/dist-packages (from jupyter->nlpia) (6.1.0)\n","Requirement already satisfied: nbconvert in
/usr/local/lib/python3.10/dist-packages (from jupyter->nlpia) (6.5.4)\n","Requirement already satisfied: ipykernel in /usr/local/lib/python3.10/dist-
packages (from jupyter->nlpia) (5.5.6)\n","Requirement already satisfied: ipywidgets in /usr/local/lib/python3.10/dist-packages (from jupyter-
>nlpia) (7.7.1)\n","Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia)
(1.0.7)\n","Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia) (0.11.0)\n","Requirement
already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia) (4.39.3)\n","Requirement already satisfied:
kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia) (1.4.4)\n","Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia) (23.1)\n","Requirement already satisfied: pillow>=6.2.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia) (8.4.0)\n","Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia) (3.0.9)\n","Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->nlpia) (2.8.2)\n","Requirement already satisfied: click in /usr/local/lib/python3.10/dist-
packages (from nltk->nlpia) (8.1.3)\n","Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk->nlpia)
(1.2.0)\n","Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->nlpia)
(2022.7.1)\n","Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.10/dist-packages (from pandas-datareader->nlpia)
(2.27.1)\n","Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly->nlpia) (8.2.2)\n","Collecting
coverage (from pugnlp->nlpia)\n"," Downloading coverage-7.2.7-cp310-cp310-
manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (228 kB)\n","\u001b[2K
\u001b[90m━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━\u001b[0m \u001b[32m228.7/228.7
kB\u001b[0m \u001b[31m24.3 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hCollecting fuzzywuzzy (from pugnlp->nlpia)\n","
Downloading fuzzywuzzy-0.18.0-py2.py3-none-any.whl (18 kB)\n","Requirement already satisfied: pip in /usr/local/lib/python3.10/dist-packages
(from pugnlp->nlpia) (23.1.2)\n","Requirement already satisfied: python-slugify in /usr/local/lib/python3.10/dist-packages (from pugnlp->nlpia)
(8.0.1)\n","Requirement already satisfied: wheel in /usr/local/lib/python3.10/dist-packages (from pugnlp->nlpia) (0.40.0)\n","Collecting
Levenshtein==0.21.0 (from python-Levenshtein->nlpia)\n"," Downloading Levenshtein-0.21.0-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (174 kB)\n","\u001b[2K
\u001b[90m━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━\u001b[0m \u001b[32m174.1/174.1
kB\u001b[0m \u001b[31m21.2 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hCollecting rapidfuzz<4.0.0,>=2.3.0 (from
Levenshtein==0.21.0->python-Levenshtein->nlpia)\n"," Downloading rapidfuzz-3.1.0-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.0 MB)\n","\u001b[2K
\u001b[90m━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━\u001b[0m \u001b[32m3.0/3.0
MB\u001b[0m \u001b[31m40.1 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hRequirement already satisfied:
threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->nlpia) (3.1.0)\n","Requirement already satisfied: spacy-
legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (3.0.12)\n","Requirement already satisfied: spacy-
loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (1.0.4)\n","Requirement already satisfied:
murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (1.0.9)\n","Requirement already satisfied:
cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (2.0.7)\n","Requirement already satisfied:
preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (3.0.8)\n","Requirement already satisfied:
thinc<8.2.0,>=8.1.8 in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (8.1.9)\n","Requirement already satisfied: wasabi<1.2.0,>=0.9.1
in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (1.1.1)\n","Requirement already satisfied: srsly<3.0.0,>=2.4.3 in
/usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (2.4.6)\n","Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in
/usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (2.0.8)\n","Requirement already satisfied: typer<0.8.0,>=0.3.0 in
/usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (0.7.0)\n","Requirement already satisfied: pathy>=0.10.0 in
/usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (0.10.1)\n","Requirement already satisfied: pydantic!=1.8,!=1.8.1,<1.11.0,>=1.7.4 in
/usr/local/lib/python3.10/dist-packages (from spacy->nlpia) (1.10.7)\n","Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-
packages (from spacy->nlpia) (3.1.2)\n","Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia)
(67.7.2)\n","Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from spacy->nlpia)
(3.3.0)\n","Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(1.4.0)\n","Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(1.6.3)\n","Requirement already satisfied: flatbuffers>=2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(23.3.3)\n","Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(0.4.0)\n","Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(0.2.0)\n","Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(1.54.0)\n","Requirement already satisfied: jax>=0.3.15 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(0.4.10)\n","Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(16.0.0)\n","Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia)
(3.3.0)\n","Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in
/usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia) (3.20.3)\n","Requirement already satisfied: tensorboard<2.13,>=2.12 in
/usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia) (2.12.2)\n","Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in
```

/usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia) (2.12.0)\n","Requirement already satisfied: termcolor>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia) (2.3.0)\n","Requirement already satisfied: typing-extensions>=3.6.6 in
/usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia) (4.5.0)\n","Requirement already satisfied: wrapt<1.15,>=1.11.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia) (1.14.1)\n","Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1
in /usr/local/lib/python3.10/dist-packages (from tensorflow->nlpia) (0.32.0)\n","Requirement already satisfied: ml-dtypes>=0.1.0 in
/usr/local/lib/python3.10/dist-packages (from jax>=0.3.15->tensorflow->nlpia) (0.1.0)\n","Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->pandas-datareader->nlpia) (1.26.15)\n","Requirement already satisfied:
certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->pandas-datareader->nlpia) (2022.12.7)\n","Requirement
already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->pandas-datareader->nlpia)
(2.0.12)\n","Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->pandas-datareader->nlpia) (3.4)\n","Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensorflow->nlpia) (2.17.3)\n","Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in /usr/local/lib/python3.10/dist-packages (from
tensorboard<2.13,>=2.12->tensorflow->nlpia) (1.0.0)\n","Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensorflow->nlpia) (3.4.3)\n","Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0
in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensorflow->nlpia) (0.7.0)\n","Requirement already satisfied:
tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensorflow->nlpia)
(1.8.1)\n","Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.13,>=2.12->tensorflow->nlpia) (2.3.0)\n","Requirement already satisfied: blis<0.8.0,>=0.7.8 in /usr/local/lib/python3.10/dist-packages (from
thinc<8.2.0,>=8.1.8->spacy->nlpia) (0.7.9)\n","Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.10/dist-packages
(from thinc<8.2.0,>=8.1.8->spacy->nlpia) (0.0.4)\n","Requirement already satisfied: ipython-genutils in /usr/local/lib/python3.10/dist-packages
(from ipykernel->jupyter->nlpia) (0.2.0)\n","Requirement already satisfied: ipython>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from
ipykernel->jupyter->nlpia) (7.34.0)\n","Requirement already satisfied: traitlets>=4.1.0 in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter->nlpia) (5.7.1)\n","Requirement already satisfied: jupyter-client in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter->nlpia)
(6.1.12)\n","Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter->nlpia)
(6.3.1)\n","Requirement already satisfied: widgetsnbextension~=3.6.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets->jupyter->nlpia)
(3.6.4)\n","Requirement already satisfied: jupyterlab-widgets>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets->jupyter->nlpia)
(3.0.7)\n","Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->spacy->nlpia)
(2.1.2)\n","Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from jupyter-console->jupyter->nlpia) (3.0.38)\n","Requirement already satisfied: pygments in /usr/local/lib/python3.10/dist-packages (from jupyter-console->jupyter->nlpia) (2.14.0)\n","Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia) (4.11.2)\n","Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(6.0.0)\n","Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(0.7.1)\n","Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(0.4)\n","Requirement already satisfied: jupyter-core>=4.7 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(5.3.0)\n","Requirement already satisfied: jupyterlab-pygments in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(0.2.2)\n","Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(0.8.4)\n","Requirement already satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(0.7.4)\n","Requirement already satisfied: nbformat>=5.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(5.8.0)\n","Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(1.5.0)\n","Requirement already satisfied: tinycss2 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter->nlpia)
(1.2.1)\n","Requirement already satisfied: pyzmq>=17 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter->nlpia)
(23.2.1)\n","Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter->nlpia)
(21.3.0)\n","Requirement already satisfied: nest-asyncio>=1.5 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter->nlpia)
(1.5.6)\n","Requirement already satisfied: Send2Trash>=1.8.0 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter->nlpia)
(1.8.0)\n","Requirement already satisfied: terminado>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter->nlpia)
(0.17.1)\n","Requirement already satisfied: prometheus-client in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter->nlpia)
(0.16.0)\n","Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.10/dist-packages (from python-slugify->pugnlp->nlpia)
(1.3)\n","Collecting qtpy>=2.0.1 (from qtconsole->jupyter->nlpia)\n"," Downloading QtPy-2.3.1-py3-none-any.whl (84 kB)\n","\u001b[2K
\u001b[90m━━\u001b[0m \u001b[32m84.9/84.9
kB\u001b[0m \u001b[31m10.0 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hRequirement already satisfied:
cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow->nlpia)
(5.3.0)\n","Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow->nlpia) (0.3.0)\n","Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages
(from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow->nlpia) (4.9)\n","Requirement already satisfied: requests-oauthlib>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib<1.1,>=0.5->tensorboard<2.13,>=2.12->tensorflow->nlpia) (1.3.1)\n","Collecting
jedi>=0.16 (from ipython>=5.0.0->ipykernel->jupyter->nlpia)\n"," Downloading jedi-0.18.2-py2.py3-none-any.whl (1.6 MB)\n","\u001b[2K
\u001b[90m━━\u001b[0m \u001b[32m1.6/1.6
MB\u001b[0m \u001b[31m9.8 MB/s\u001b[0m eta \u001b[36m0:00:00\u001b[0m\n","\u001b[?25hRequirement already satisfied: decorator in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->jupyter->nlpia) (4.4.2)\n","Requirement already satisfied: pickleshare in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->jupyter->nlpia) (0.7.5)\n","Requirement already satisfied: backcall in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->jupyter->nlpia) (0.2.0)\n","Requirement already satisfied: matplotlib-inline in /usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->jupyter->nlpia) (0.1.6)\n","Requirement already satisfied:
pexpect>4.3 in /usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->jupyter->nlpia) (4.8.0)\n","Requirement already satisfied:
platformdirs>=2.5 in /usr/local/lib/python3.10/dist-packages (from jupyter-core>=4.7->nbconvert->jupyter->nlpia) (3.3.0)\n","Requirement already
satisfied: fastjsonschema in /usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbconvert->jupyter->nlpia) (2.16.3)\n","Requirement
already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbconvert->jupyter->nlpia)
(4.3.3)\n","Requirement already satisfied: wcwidth in /usr/local/lib/python3.10/dist-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->jupyter-console->jupyter->nlpia) (0.2.6)\n","Requirement already satisfied: ptyprocess in /usr/local/lib/python3.10/dist-packages (from
terminado>=0.8.3->notebook->jupyter->nlpia) (0.7.0)\n","Requirement already satisfied: argon2-cffi-bindings in /usr/local/lib/python3.10/dist-packages (from argon2-cffi->notebook->jupyter->nlpia) (21.2.0)\n","Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->nbconvert->jupyter->nlpia) (2.4.1)\n","Requirement already satisfied: parso<0.9.0,>=0.8.0 in
/usr/local/lib/python3.10/dist-packages (from jedi>=0.16->ipython>=5.0.0->ipykernel->jupyter->nlpia) (0.8.3)\n","Requirement already satisfied:
attrs>=17.4.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat>=5.1->nbconvert->jupyter->nlpia)
(23.1.0)\n","Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in /usr/local/lib/python3.10/dist-packages (from
jsonschema>=2.6->nbformat>=5.1->nbconvert->jupyter->nlpia) (0.19.3)\n","Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in
/usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow->nlpia)
(0.5.0)\n","Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<1.1,>=0.5->tensorboard<2.13,>=2.12->tensorflow->nlpia) (3.2.2)\n","Requirement already satisfied: cffi>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from argon2-cffi-bindings->argon2-cffi->notebook->jupyter->nlpia) (1.15.1)\n","Requirement already
satisfied: pycparser in /usr/local/lib/python3.10/dist-packages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook->jupyter->nlpia)
(2.21)\n","Installing collected packages: fuzzywuzzy, rapidfuzz, qtpy, pypandoc, jedi, html2text, coverage, Levenshtein, python-Levenshtein,
qtconsole, jupyter, pugnlp, nlpia\n","Successfully installed Levenshtein-0.21.0 coverage-7.2.7 fuzzywuzzy-0.18.0 html2text-2020.1.16 jedi-0.18.2
jupyter-1.0.0 nlpia-0.5.2 pugnlp-0.2.6 pypandoc-1.11 python-Levenshtein-0.21.0 qtconsole-5.4.3 qtpy-2.3.1 rapidfuzz-3.1.0\n"]}]},
{"cell_type":"code","source":["from nlpia.data.loaders import get_data # noqa\n","# movies = get_data('hutto_movies')"],"metadata":
{"id":"gtpn71k7fl10"},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["movies.head().round(2)"],"metadata":

{"id":"xE5xD5bLfdoT","executionInfo":{"status":"aborted","timestamp":1685957790034,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["movies.describe().round(2)\n","# movies were rated on a scale from -4 to +4"],"metadata":{"id":"Jx9oK07CgBQS","executionInfo":{"status":"aborted","timestamp":1685957790035,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"markdown","source":["Now let’s tokenize all those movie review texts to create a bag of words for each one.\n","\n","You’ll put them all into a Pandas DataFrame:"],"metadata":{"id":"klF9uxaggHNq"}},{"cell_type":"code","source":["import pandas as pd\n","\n","# This line helps display wide DataFrames in the console\n","# so they look prettier.\n","pd.set_option('display.width', 75)\n","\n","\n","# NLTK’s casual_tokenize can handle emoticons, unusual punctuation, and slang\n","# better than Treebank Word Tokenizer.\n","from nltk.tokenize import casual_tokenize\n","bags_of_words = []"],"metadata":{"id":"sIxnMFeHgXqB","executionInfo":{"status":"aborted","timestamp":1685957790036,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["# The Python built-in Counter takes a list of objects and counts them,\n","# returning a dictionary where the keys are the objects (tokens in your\n","# case) and the values are the integer counts of those objects.\n","from collections import Counter\n","\n","for text in movies.text:\n"," bags_of_words.append(Counter(casual_tokenize(text)))\n"],"metadata":{"id":"hom2k6pGguxK","executionInfo":{"status":"aborted","timestamp":1685957790036,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["# The from_records() DataFrame constructor takes a\n","# sequence of dictionaries. It creates columns for all the\n","# keys, and the values are added to the table in the\n","# appropriate columns, filling missing values with NaN.\n","df_bows = pd.DataFrame.from_records(bags_of_words)"],"metadata":{"id":"azKuTgLjg94Z","executionInfo":{"status":"aborted","timestamp":1685957790037,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["# Numpy and Pandas can only represent NaNs in float\n","# objects, so once you fill all the NaNs with zeros you\n","# can convert the DataFrame to integers, which are\n","# much more compact (in memory and to display).\n","df_bows = df_bows.fillna(0).astype(int)"],"metadata":{"id":"cstX0EjQhF3J","executionInfo":{"status":"aborted","timestamp":1685957790037,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["# A bag-of-words table can grow quite large quickly, especially when you don’t \n","# use case normalization, stop word filters, stemming, and lemmatization. \n","# Try inserting some of these dimension reducers here and see how they affect \n","# your pipeline.\n","df_bows.shape"],"metadata":{"id":"mjLWhltghSJZ","executionInfo":{"status":"aborted","timestamp":1685957790037,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["df_bows.head()"],"metadata":{"id":"z82-ek2PhzOz","executionInfo":{"status":"aborted","timestamp":1685957790038,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["df_bows.head()[list(bags_of_words[0].keys())]"],"metadata":{"id":"GqiNMpFth3W6","executionInfo":{"status":"aborted","timestamp":1685957790038,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"markdown","source":["Now you have all the data that a Naive Bayes model needs to find the keywords that\n","predict sentiment from natural language text:"],"metadata":{"id":"AOZd8D5Bh5qK"}},{"cell_type":"markdown","source":["Naive Bayes models are classifiers, so you need to convert your output variable (sentiment float) to a discrete label (integer, string, or bool)."],"metadata":{"id":"MrXsyCLViH-o"}},{"cell_type":"code","source":["from sklearn.naive_bayes import MultinomialNB\n","nb = MultinomialNB()\n","nb = nb.fit(df_bows, movies.sentiment > 0)"],"metadata":{"id":"7xCqhzM5iKhy","executionInfo":{"status":"aborted","timestamp":1685957790039,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["# Convert your binary classification variable (0 or 1) to -4 or 4 so you\n","# can compare it to the “ground truth” sentiment. Use nb.predict_proba to\n","# get a continuous value.\n","movies['predicted_sentiment'] = nb.predict(df_bows) * 8 - 4"],"metadata":{"id":"Sejnuv7hiQP6","executionInfo":{"status":"aborted","timestamp":1685957790039,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["# The average absolute value of the prediction error (mean\n","# absolute error or MAE) is 2.4.\n","movies['error'] = (movies.predicted_sentiment - movies.sentiment).abs()\n","round(movies.error.mean(), 1) "],"metadata":{"id":"ap-g-t1PidBd","executionInfo":{"status":"aborted","timestamp":1685957790039,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["movies['sentiment_ispositive'] = (movies.sentiment > 0).astype(int)\n","movies['predicted_ispositive'] = (movies.predicted_sentiment > 0).astype(int)\n","movies['sentiment predicted_sentiment sentiment_ispositive predicted_ispositive'.split()].head(8)"],"metadata":{"id":"1swHZswkj9la","executionInfo":{"status":"aborted","timestamp":1685957790040,"user_tz":-330,"elapsed":21,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"code","source":["(movies.predicted_ispositive == movies.sentiment_ispositive).sum() / len(movies)"],"metadata":{"id":"pg-T5rehk12C","executionInfo":{"status":"aborted","timestamp":1685957790041,"user_tz":-330,"elapsed":22,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]},{"cell_type":"markdown","source":["**You got the “thumbs up” rating correct 93% of the time.**"],"metadata":{"id":"HsePSzqwl256"}},{"cell_type":"code","source":[],"metadata":{"id":"PVgB8OHwTjKQ","executionInfo":{"status":"aborted","timestamp":1685957790042,"user_tz":-330,"elapsed":23,"user":{"displayName":"Dr. Muneendra Ojha","userId":"06154531826228794240"}}},"execution_count":null,"outputs":[]}]}