```
# installation of spacy
!pip install spacy
```

```
Requirement already satisfied: spacy in c:\users\dell\anaconda3\lib\
site-packages (3.5.3)
Requirement already satisfied: pathy>=0.10.0 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (0.10.1)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (2.0.7)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (3.0.8)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in c:\users\
dell\anaconda3\lib\site-packages (from spacy) (2.0.8)
Requirement already satisfied: smart-open<7.0.0,>=5.2.1 in c:\users\
dell\anaconda3\lib\site-packages (from spacy) (6.3.0)
Requirement already satisfied: packaging>=20.0 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (20.9)
Requirement already satisfied: numpy>=1.15.0 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (1.23.5)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (1.1.1)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in c:\
users\dell\anaconda3\lib\site-packages (from spacy) (3.0.12)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in c:\users\
dell\anaconda3\lib\site-packages (from spacy) (3.3.0)
Requirement already satisfied: jinja2 in c:\users\dell\anaconda3\lib\
site-packages (from spacy) (2.11.3)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in c:\users\
dell\anaconda3\lib\site-packages (from spacy) (1.0.9)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in c:\users\
dell\anaconda3\lib\site-packages (from spacy) (2.31.0)
Requirement already satisfied: typer<0.8.0,>=0.3.0 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (0.7.0)
Requirement already satisfied: thinc<8.2.0,>=8.1.8 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (8.1.10)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (4.66.1)
Requirement already satisfied: setuptools in c:\users\dell\anaconda3\
lib\site-packages (from spacy) (52.0.0.post20210125)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in c:\users\dell\
anaconda3\lib\site-packages (from spacy) (2.4.6)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in c:\
users\dell\anaconda3\lib\site-packages (from spacy) (1.0.4)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<1.11.0,>=1.7.4
in c:\users\dell\anaconda3\lib\site-packages (from spacy) (1.10.8)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\dell\
anaconda3\lib\site-packages (from packaging>=20.0->spacy) (2.4.7)
Requirement already satisfied: typing-extensions>=4.2.0 in c:\users\
dell\anaconda3\lib\site-packages (from pydantic!=1.8,!
=1.8.1,<1.11.0,>=1.7.4->spacy) (4.6.1)
```

```
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\dell\
anaconda3\lib\site-packages (from requests<3.0.0,>=2.13.0->spacy)
(1.26.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\dell\
anaconda3\lib\site-packages (from requests<3.0.0,>=2.13.0->spacy)
(2020.12.5)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\
dell\anaconda3\lib\site-packages (from requests<3.0.0,>=2.13.0->spacy)
(3.2.0)
Requirement already satisfied: idna<4,>=2.5 in c:\users\dell\
anaconda3\lib\site-packages (from requests<3.0.0,>=2.13.0->spacy)
(2.10)
Requirement already satisfied: confection<1.0.0,>=0.0.1 in c:\users\
dell\anaconda3\lib\site-packages (from thinc<8.2.0,>=8.1.8->spacy)
(0.0.4)
Requirement already satisfied: blis<0.8.0,>=0.7.8 in c:\users\dell\
anaconda3\lib\site-packages (from thinc<8.2.0,>=8.1.8->spacy) (0.7.9)
Requirement already satisfied: colorama in c:\users\dell\anaconda3\
lib\site-packages (from tqdm<5.0.0,>=4.38.0->spacy) (0.4.6)
Requirement already satisfied: click<9.0.0,>=7.1.1 in c:\users\dell\
anaconda3\lib\site-packages (from typer<0.8.0,>=0.3.0->spacy) (7.1.2)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\dell\
anaconda3\lib\site-packages (from jinja2->spacy) (1.1.1)
```

```
import spacy

spacy.cli.download("en_core_web_sm") # small, medium - md , large - lg
```

```
✔ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
```

```
nlp = spacy.load("en_core_web_sm")

# Tokenization


text = "Hi my name is munira fatima."
print(text)
```

```
Hi my name is munira fatima.
```

```
print(type(text))
```

```
<class 'str'>
```

```
doc = nlp(text)
print(doc)
```

```
Hi my name is munira fatima.
```

```
print(type(doc))
```

```
<class 'spacy.tokens.doc.Doc'>
```

```
tokens = [word for word in text.split()]
print(tokens)
```

```
['Hi', 'my', 'name', 'is', 'munira', 'fatima.']
```

```
import nltk
```

```
from nltk import word_tokenize, sent_tokenize
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\DELL\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

```
True
```

```
print(word_tokenize(text))
```

```
['Hi', 'my', 'name', 'is', 'munira', 'fatima', '.']
```

```
print(doc)
review = "Levis is a nice brand"
```

```
doc = nlp(review)
print(doc)
```

```
Levis is a nice brand
Levis is a nice brand
```

```
filtered_doc = []
for token in doc:
    if not token.is_stop:
        filtered_doc.append(token)
```

```
print(filtered_doc)
```

```
[Levis, nice, brand]
```

```
# POS - Parts of Speech
```

```
print(doc)
```

```
pos_text = [token.pos_ for token in doc]
print(pos_text)
```

```
Levis is a nice brand
['PROPN', 'AUX', 'DET', 'ADJ', 'NOUN']
```

```
pos_dic = {}
```

```
for token in doc:
```

```
    pos_dic[token] = token.pos_
print(pos_dic)
```

```
{Levis: 'PROPN', is: 'AUX', a: 'DET', nice: 'ADJ', brand: 'NOUN'}
```

```
# tokenization - get rid of stopwords --> token.is_stop
# Part of Speech - token.pos_

l1 = [token.pos_ for token in doc]
l2 = [token for token in doc]
print(dict(zip(l2,l1)))
```

```
{Levis: 'PROPN', is: 'AUX', a: 'DET', nice: 'ADJ', brand: 'NOUN'}
```

```
list1 = [1,2,3,4]
diction = {val : str(val) for val in list1}
print(diction)
```

```
{1: '1', 2: '2', 3: '3', 4: '4'}
```

```
result = {token : token.pos_ for token in doc}
print(result)
```

```
{Levis: 'PROPN', is: 'AUX', a: 'DET', nice: 'ADJ', brand: 'NOUN'}
```

```
# lemmatization
# token.is_stop
# token.pos_
# token.lemma_


sentence = "Sunsilk is a nice shampoo eat ate eaten eating eats"
doc = nlp(sentence)
filter_sent = [token for token in doc if not token.is_stop]
print(filter_sent)

lemma_text = [token.lemma_ for token in filter_sent]
print(lemma_text)
```

```
[Sunsilk, nice, shampoo, eat, ate, eaten, eating, eats]
['sunsilk', 'nice', 'shampoo', 'eat', 'ate', 'eat', 'eat', 'eat']
```

```
remove_duplicates = []
for word in lemma_text:
    if word not in remove_duplicates:
        remove_duplicates.append(word)
print(remove_duplicates)
```

```
['sunsilk', 'nice', 'shampoo', 'eat', 'ate']
```

```
print(list(set(lemma_text)))
```

```
['sunsilk', 'eat', 'shampoo', 'nice', 'ate']
```

```
# Name Entity Recognize - token.label_
# lemmatization - token.lemma_

text = "My name is munira fatima and I am living in India along with
my parents and working for Microsoft"

doc = nlp(text)
print(doc)
```

```
My name is munira fatima and I am living in India along with my
parents and working for Microsoft
```

```
for ent in doc.ents:
    print(ent, ent.label_)
```

```
munira fatima PERSON
India GPE
Microsoft ORG
```

```
print(doc.ents)
```

```
(munira fatima, India, Microsoft)
```

```
print(type(doc))
```

```
<class 'spacy.tokens.doc.Doc'>
```

```
help(doc)
```

```
Help on Doc object:

class Doc(builtins.object)
 |  Doc(Vocab vocab, words=None, spaces=None, user_data=None, *,
tags=None, pos=None, morphs=None, lemmas=None, heads=None, deps=None,
sent_starts=None, ents=None)
 |  A sequence of Token objects. Access sentences and named entities,
export
 |      annotations to numpy arrays, losslessly serialize to
compressed binary
 |      strings. The `Doc` object holds an array of `TokenC` structs.
The
 |      Python-level `Token` and `Span` objects are views of this
array, i.e.
 |      they don't own the data themselves.
 |
 |      EXAMPLE:
 |          Construction 1
 |          >>> doc = nlp(u'Some text')
 |
 |          Construction 2
 |          >>> from spacy.tokens import Doc
 |          >>> doc = Doc(nlp.vocab, words=["hello", "world", "!"],
```

```
spaces=[True, False, False])
 |
 |      DOCS: https://spacy.io/api/doc
 |
 |  Methods defined here:
 |
 |  __bytes__(...)
 |      Doc.__bytes__(self)
 |
 |  __getitem__(...)
 |      Get a `Token` or `Span` object.
 |
 |      i (int or tuple) The index of the token, or the slice of the
document
 |          to get.
 |      RETURNS (Token or Span): The token at `doc[i]]`, or the span
at
 |          `doc[start : end]`.
 |
 |      EXAMPLE:
 |          >>> doc[i]
 |          Get the `Token` object at position `i`, where `i` is an
integer.
 |          Negative indexing is supported, and follows the usual
Python
 |          semantics, i.e. `doc[-2]` is `doc[len(doc) - 2]`.
 |
 |          >>> doc[start : end]]
 |          Get a `Span` object, starting at position `start` and
ending at
 |          position `end`, where `start` and `end` are token indices.
For
 |          instance, `doc[2:5]` produces a span consisting of tokens
2, 3 and
 |          4. Stepped slices (e.g. `doc[start : end : step]`) are not
 |          supported, as `Span` objects must be contiguous (cannot
have gaps).
 |          You can use negative indices and open-ended ranges, which
have
 |          their normal Python semantics.
 |
 |      DOCS: https://spacy.io/api/doc#getitem
 |
 |  __init__(...)
 |      Create a Doc object.
 |
 |      vocab (Vocab): A vocabulary object, which must match any
models you
 |          want to use (e.g. tokenizer, parser, entity recognizer).
```

```
 |      words (Optional[List[Union[str, int]]]): A list of unicode
strings or
 |          hash values to add to the document as words. If `None`,
defaults to
 |          empty list.
 |      spaces (Optional[List[bool]]): A list of boolean values, of
the same
 |          length as `words`. `True` means that the word is followed
by a space,
 |          `False` means it is not. If `None`, defaults to
`[True]*len(words)`
 |      user_data (dict or None): Optional extra data to attach to the
Doc.
 |      tags (Optional[List[str]]): A list of unicode strings, of the
same
 |          length as words, to assign as token.tag. Defaults to None.
 |      pos (Optional[List[str]]): A list of unicode strings, of the
same
 |          length as words, to assign as token.pos. Defaults to None.
 |      morphs (Optional[List[str]]): A list of unicode strings, of
the same
 |          length as words, to assign as token.morph. Defaults to
None.
 |      lemmas (Optional[List[str]]): A list of unicode strings, of
the same
 |          length as words, to assign as token.lemma. Defaults to
None.
 |      heads (Optional[List[int]]): A list of values, of the same
length as
 |          words, to assign as heads. Head indices are the position
of the
 |          head in the doc. Defaults to None.
 |      deps (Optional[List[str]]): A list of unicode strings, of the
same
 |          length as words, to assign as token.dep. Defaults to None.
 |      sent_starts (Optional[List[Union[bool, int, None]]]): A list
of values,
 |          of the same length as words, to assign as
token.is_sent_start. Will
 |          be overridden by heads if heads is provided. Defaults to
None.
 |      ents (Optional[List[str]]): A list of unicode strings, of the
same
 |          length as words, as IOB tags to assign as token.ent_iob
and
 |          token.ent_type. Defaults to None.
 |
 |      DOCS: https://spacy.io/api/doc#init
 |
```

```
 |  __iter__(...)
 |      Iterate over `Token`  objects, from which the annotations can
be
 |      easily accessed. This is the main way of accessing `Token`
objects,
 |      which are the main way annotations are accessed from Python.
If faster-
 |      than-Python speeds are required, you can instead access the
annotations
 |      as a numpy array, or access the underlying C data directly
from Cython.
 |
 |      DOCS: https://spacy.io/api/doc#iter
 |
 |  __len__(...)
 |      The number of tokens in the document.
 |
 |      RETURNS (int): The number of tokens in the document.
 |
 |      DOCS: https://spacy.io/api/doc#len
 |
 |  __reduce__ = __reduce_cython__(...)
 |      Doc.__reduce_cython__(self)
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__ = __setstate_cython__(...)
 |      Doc.__setstate_cython__(self, __pyx_state)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  __unicode__(...)
 |      Doc.__unicode__(self)
 |
 |  char_span(...)
 |      Doc.char_span(self, int start_idx, int end_idx, label=0,
kb_id=0, vector=None, alignment_mode='strict', span_id=0)
 |      Create a `Span` object from the slice
 |              `doc.text[start_idx : end_idx]`. Returns None if no
valid `Span` can be
 |              created.
 |
 |              doc (Doc): The parent document.
 |              start_idx (int): The index of the first character of
the span.
 |              end_idx (int): The index of the first character after
the span.
```

```
 |               label (Union[int, str]): A label to attach to the
Span, e.g. for
 |                   named entities.
 |               kb_id (Union[int, str]):  An ID from a KB to capture
the meaning of a
 |                   named entity.
 |               vector (ndarray[ndim=1, dtype='float32']): A meaning
representation of
 |                   the span.
 |               alignment_mode (str): How character indices are
aligned to token
 |                   boundaries. Options: "strict" (character indices
must be aligned
 |                   with token boundaries), "contract" (span of all
tokens completely
 |                   within the character span), "expand" (span of all
tokens at least
 |                   partially covered by the character span). Defaults
to "strict".
 |               span_id (Union[int, str]): An identifier to associate
with the span.
 |               RETURNS (Span): The newly constructed object.
 |
 |               DOCS: https://spacy.io/api/doc#char_span
 |
 |  copy(...)
 |      Doc.copy(self)
 |
 |  count_by(...)
 |      Doc.count_by(self, attr_id_t attr_id, exclude=None,
counts=None)
 |      Count the frequencies of a given attribute. Produces a dict of
 |               `{attribute (int): count (ints)}` frequencies, keyed
by the values of
 |               the given attribute ID.
 |
 |               attr_id (int): The attribute ID to key the counts.
 |               RETURNS (dict): A dictionary mapping attributes to
integer counts.
 |
 |               DOCS: https://spacy.io/api/doc#count_by
 |
 |  extend_tensor(...)
 |      Doc.extend_tensor(self, tensor)
 |      Concatenate a new tensor onto the doc.tensor object.
 |
 |               The doc.tensor attribute holds dense feature vectors
 |               computed by the models in the pipeline. Let's say a
 |               document with 30 words has a tensor with 128
```

```
dimensions
 |              per word. doc.tensor.shape will be (30, 128). After
 |              calling doc.extend_tensor with an array of shape (30,
64),
 |              doc.tensor == (30, 192).
 |
 |  from_array(...)
 |      Doc.from_array(self, attrs, array)
 |      Load attributes from a numpy array. Write to a `Doc` object,
from an
 |              `(M, N)` array of attributes.
 |
 |              attrs (list) A list of attribute ID ints.
 |              array (numpy.ndarray[ndim=2, dtype='int32']): The
attribute values.
 |              RETURNS (Doc): Itself.
 |
 |              DOCS: https://spacy.io/api/doc#from_array
 |
 |  from_bytes(...)
 |      Doc.from_bytes(self, bytes_data, *, exclude=tuple())
 |      Deserialize, i.e. import the document contents from a binary
string.
 |
 |              data (bytes): The string to load from.
 |              exclude (list): String names of serialization fields
to exclude.
 |              RETURNS (Doc): Itself.
 |
 |              DOCS: https://spacy.io/api/doc#from_bytes
 |
 |  from_dict(...)
 |      Doc.from_dict(self, msg, *, exclude=tuple())
 |      Deserialize, i.e. import the document contents from a binary
string.
 |
 |              data (bytes): The string to load from.
 |              exclude (list): String names of serialization fields
to exclude.
 |              RETURNS (Doc): Itself.
 |
 |              DOCS: https://spacy.io/api/doc#from_dict
 |
 |  from_disk(...)
 |      Doc.from_disk(self, path, *, exclude=tuple())
 |      Loads state from a directory. Modifies the object in place and
 |              returns it.
 |
 |              path (str / Path): A path to a directory. Paths may be
```

```
either
 |                  strings or `Path`-like objects.
 |              exclude (list): String names of serialization fields
to exclude.
 |              RETURNS (Doc): The modified `Doc` object.
 |
 |              DOCS: https://spacy.io/api/doc#from_disk
 |
 |  from_json(...)
 |      Doc.from_json(self, doc_json, *, validate=False)
 |      Convert a JSON document generated by Doc.to_json() to a Doc.
 |
 |              doc_json (Dict): JSON representation of doc object to
load.
 |              validate (bool): Whether to validate `doc_json`
against the expected schema.
 |                  Defaults to False.
 |              RETURNS (Doc): A doc instance corresponding to the
specified JSON representation.
 |
 |  get_lca_matrix(...)
 |      Doc.get_lca_matrix(self)
 |      Calculates a matrix of Lowest Common Ancestors (LCA) for a
given
 |              `Doc`, where LCA[i, j] is the index of the lowest
common ancestor among
 |              token i and j.
 |
 |              RETURNS (np.array[ndim=2, dtype=numpy.int32]): LCA
matrix with shape
 |                  (n, n), where n = len(self).
 |
 |              DOCS: https://spacy.io/api/doc#get_lca_matrix
 |
 |  has_annotation(...)
 |      Doc.has_annotation(self, attr, *, require_complete=False)
 |      Check whether the doc contains annotation on a token
attribute.
 |
 |              attr (Union[int, str]): The attribute string name or
int ID.
 |              require_complete (bool): Whether to check that the
attribute is set on
 |                  every token in the doc.
 |              RETURNS (bool): Whether annotation is present.
 |
 |              DOCS: https://spacy.io/api/doc#has_annotation
 |
 |  retokenize(...)
```

```
 |      Doc.retokenize(self)
 |      Context manager to handle retokenization of the Doc.
 |              Modifications to the Doc's tokenization are stored,
and then
 |              made all at once when the context manager exits. This
is
 |              much more efficient, and less error-prone.
 |
 |              All views of the Doc (Span and Token) created before
the
 |              retokenization are invalidated, although they may
accidentally
 |              continue to work.
 |
 |              DOCS: https://spacy.io/api/doc#retokenize
 |              USAGE: https://spacy.io/usage/linguistic-
features#retokenization
 |
 |  set_ents(...)
 |      Doc.set_ents(self, entities, *, blocked=None, missing=None,
outside=None, default=SetEntsDefault.outside)
 |      Set entity annotation.
 |
 |              entities (List[Span]): Spans with labels to set as
entities.
 |              blocked (Optional[List[Span]]): Spans to set as
'blocked' (never an
 |                  entity) for spacy's built-in NER component. Other
components may
 |                  ignore this setting.
 |              missing (Optional[List[Span]]): Spans with
missing/unknown entity
 |                  information.
 |              outside (Optional[List[Span]]): Spans outside of
entities (O in IOB).
 |              default (str): How to set entity annotation for tokens
outside of any
 |                  provided spans. Options: "blocked", "missing",
"outside" and
 |                  "unmodified" (preserve current state). Defaults to
"outside".
 |
 |  similarity(...)
 |      Doc.similarity(self, other)
 |      Make a semantic similarity estimate. The default estimate is
cosine
 |              similarity using an average of word vectors.
 |
 |              other (object): The object to compare with. By
```

```
default, accepts `Doc`,
 |                  `Span`, `Token` and `Lexeme` objects.
 |              RETURNS (float): A scalar similarity score. Higher is
more similar.
 |
 |              DOCS: https://spacy.io/api/doc#similarity
 |
 |  to_array(...)
 |      Doc.to_array(self, py_attr_ids) -> ndarray
 |      Export given token attributes to a numpy `ndarray`.
 |              If `attr_ids` is a sequence of M attributes, the
output array will be
 |              of shape `(N, M)`, where N is the length of the `Doc`
(in tokens). If
 |              `attr_ids` is a single attribute, the output shape
will be (N,). You
 |              can specify attributes by integer ID (e.g.
spacy.attrs.LEMMA) or
 |              string name (e.g. 'LEMMA' or 'lemma').
 |
 |              py_attr_ids (list[]): A list of attributes (int IDs or
string names).
 |              RETURNS (numpy.ndarray[long, ndim=2]): A feature
matrix, with one row
 |                  per word, and one column per attribute indicated
in the input
 |                  `attr_ids`.
 |
 |              EXAMPLE:
 |                  >>> from spacy.attrs import LOWER, POS, ENT_TYPE,
IS_ALPHA
 |                  >>> doc = nlp(text)
 |                  >>> # All strings mapped to integers, for easy
export to numpy
 |                  >>> np_array = doc.to_array([LOWER, POS, ENT_TYPE,
IS_ALPHA])
 |
 |  to_bytes(...)
 |      Doc.to_bytes(self, *, exclude=tuple())
 |      Serialize, i.e. export the document contents to a binary
string.
 |
 |              exclude (list): String names of serialization fields
to exclude.
 |              RETURNS (bytes): A losslessly serialized copy of the
`Doc`, including
 |                  all annotations.
 |
 |              DOCS: https://spacy.io/api/doc#to_bytes
```

```
 |
 |  to_dict(...)
 |      Doc.to_dict(self, *, exclude=tuple())
 |      Export the document contents to a dictionary for
serialization.
 |
 |              exclude (list): String names of serialization fields
to exclude.
 |              RETURNS (bytes): A losslessly serialized copy of the
`Doc`, including
 |                  all annotations.
 |
 |              DOCS: https://spacy.io/api/doc#to_bytes
 |
 |  to_disk(...)
 |      Doc.to_disk(self, path, *, exclude=tuple())
 |      Save the current state to a directory.
 |
 |              path (str / Path): A path to a directory, which will
be created if
 |                  it doesn't exist. Paths may be either strings or
Path-like objects.
 |              exclude (Iterable[str]): String names of serialization
fields to exclude.
 |
 |              DOCS: https://spacy.io/api/doc#to_disk
 |
 |  to_json(...)
 |      Doc.to_json(self, underscore=None)
 |      Convert a Doc to JSON.
 |
 |              underscore (list): Optional list of string names of
custom doc._.
 |              attributes. Attribute values need to be JSON-
serializable. Values will
 |              be added to an "_" key in the data, e.g. "_": {"foo":
"bar"}.
 |              RETURNS (dict): The data in JSON format.
 |
 |  to_utf8_array(...)
 |      Doc.to_utf8_array(self, int nr_char=-1)
 |      Encode word strings to utf8, and export to a fixed-width array
 |              of characters. Characters are placed into the array in
the order:
 |                  0, -1, 1, -2, etc
 |              For example, if the array is sliced array[:, :8], the
array will
 |              contain the first 4 characters and last 4 characters
of each word ---
```

```
 |              with the middle characters clipped out. The value 255
is used as a pad
 |              value.
 |
 |
----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  get_extension(...) from builtins.type
 |      Doc.get_extension(type cls, name)
 |      Look up a previously registered extension by name.
 |
 |              name (str): Name of the extension.
 |              RETURNS (tuple): A `(default, method, getter, setter)`
tuple.
 |
 |              DOCS: https://spacy.io/api/doc#get_extension
 |
 |  has_extension(...) from builtins.type
 |      Doc.has_extension(type cls, name)
 |      Check whether an extension has been registered.
 |
 |              name (str): Name of the extension.
 |              RETURNS (bool): Whether the extension has been
registered.
 |
 |              DOCS: https://spacy.io/api/doc#has_extension
 |
 |  remove_extension(...) from builtins.type
 |      Doc.remove_extension(type cls, name)
 |      Remove a previously registered extension.
 |
 |              name (str): Name of the extension.
 |              RETURNS (tuple): A `(default, method, getter, setter)`
tuple of the
 |                  removed extension.
 |
 |              DOCS: https://spacy.io/api/doc#remove_extension
 |
 |  set_extension(...) from builtins.type
 |      Doc.set_extension(type cls, name, **kwargs)
 |      Define a custom attribute which becomes available as `Doc._`.
 |
 |              name (str): Name of the attribute to set.
 |              default: Optional default value of the attribute.
 |              getter (callable): Optional getter function.
 |              setter (callable): Optional setter function.
 |              method (callable): Optional method for method
extension.
 |              force (bool): Force overwriting existing attribute.
```

```
 |
 |              DOCS: https://spacy.io/api/doc#set_extension
 |              USAGE: https://spacy.io/usage/processing-
pipelines#custom-components-attributes
 |
 |
----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate
signature.
 |
 |  from_docs(...)
 |      Doc.from_docs(docs, ensure_whitespace=True, attrs=None, *,
exclude=tuple())
 |      Concatenate multiple Doc objects to form a new one. Raises an
error
 |              if the `Doc` objects do not all share the same
`Vocab`.
 |
 |              docs (list): A list of Doc objects.
 |              ensure_whitespace (bool): Insert a space between two
adjacent docs
 |                  whenever the first doc does not end in whitespace.
 |              attrs (list): Optional list of attribute ID ints or
attribute name
 |                  strings.
 |              exclude (Iterable[str]): Doc attributes to exclude.
Supported
 |                  attributes: `spans`, `tensor`, `user_data`.
 |              RETURNS (Doc): A doc that contains the concatenated
docs, or None if no
 |                  docs were given.
 |
 |              DOCS: https://spacy.io/api/doc#from_docs
 |
 |
----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  cats
 |      cats: object
 |
 |  doc
 |
 |  ents
 |      The named entities in the document. Returns a tuple of named
entity
```

```
 |      `Span` objects, if the entity recognizer has been applied.
 |
 |      RETURNS (tuple): Entities in the document, one `Span` per
entity.
 |
 |      DOCS: https://spacy.io/api/doc#ents
 |
 |  has_unknown_spaces
 |      has_unknown_spaces: 'bool'
 |
 |  has_vector
 |      A boolean value indicating whether a word vector is associated
with
 |      the object.
 |
 |      RETURNS (bool): Whether a word vector is associated with the
object.
 |
 |      DOCS: https://spacy.io/api/doc#has_vector
 |
 |  is_nered
 |
 |  is_parsed
 |
 |  is_sentenced
 |
 |  is_tagged
 |
 |  lang
 |      RETURNS (uint64): ID of the language of the doc's vocabulary.
 |
 |  lang_
 |      RETURNS (str): Language of the doc's vocabulary, e.g. 'en'.
 |
 |  mem
 |
 |  noun_chunks
 |      Iterate over the base noun phrases in the document. Yields
base
 |      noun-phrase #[code Span] objects, if the language has a noun
chunk iterator.
 |      Raises a NotImplementedError otherwise.
 |
 |      A base noun phrase, or "NP chunk", is a noun
 |      phrase that does not permit other NPs to be nested within it –
so no
 |      NP-level coordination, no prepositional phrases, and no
relative
 |      clauses.
```

```
  |
  |      YIELDS (Span): Noun chunks in the document.
  |
  |      DOCS: https://spacy.io/api/doc#noun_chunks
  |
  |  noun_chunks_iterator
  |      noun_chunks_iterator: object
  |
  |  sentiment
  |      sentiment: 'float'
  |
  |  sents
  |      Iterate over the sentences in the document. Yields sentence
`Span`
  |      objects. Sentence spans have no label.
  |
  |      YIELDS (Span): Sentences in the document.
  |
  |      DOCS: https://spacy.io/api/doc#sents
  |
  |  spans
  |
  |  tensor
  |      tensor: object
  |
  |  text
  |      A unicode representation of the document text.
  |
  |      RETURNS (str): The original verbatim text of the document.
  |
  |  text_with_ws
  |      An alias of `Doc.text`, provided for duck-type compatibility
with
  |      `Span` and `Token`.
  |
  |      RETURNS (str): The original verbatim text of the document.
  |
  |  user_data
  |      user_data: object
  |
  |  user_hooks
  |      user_hooks: dict
  |
  |  user_span_hooks
  |      user_span_hooks: dict
  |
  |  user_token_hooks
  |      user_token_hooks: dict
  |
```

```
 |  vector
 |      A real-valued meaning representation. Defaults to an average
of the
 |      token vectors.
 |
 |      RETURNS (numpy.ndarray[ndim=1, dtype='float32']): A 1D numpy
array
 |          representing the document's semantics.
 |
 |      DOCS: https://spacy.io/api/doc#vector
 |
 |  vector_norm
 |      The L2 norm of the document's vector representation.
 |
 |      RETURNS (float): The L2 norm of the vector representation.
 |
 |      DOCS: https://spacy.io/api/doc#vector_norm
 |
 |  vocab
 |
 |
----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __pyx_vtable__ = <capsule object NULL>
```

```
print(doc)
```

```
My name is munira fatima and I am living in India along with my
parents and working for Microsoft
```

```
print(doc.lemmas)
```

```
---------------------------------------------------------------------
-----
AttributeError                            Traceback (most recent call
last)
<ipython-input-87-3c14685546d4> in <module>
----> 1 print(doc.lemmas)

AttributeError: 'spacy.tokens.doc.Doc' object has no attribute
'lemmas'
```

```
print(doc.text)
```

```
My name is munira fatima and I am living in India along with my
parents and working for Microsoft
```

```
print(doc.text_with_ws)
```

```
My name is munira fatima and I am living in India along with my
parents and working for Microsoft
```

```
print(doc.vector)
```

```
[-1.8716514e-01 -1.4162176e-02 -2.6025072e-01  2.8547725e-01
  3.0018247e-03 -4.1234933e-02  3.4674749e-01  4.6479713e-02
  3.3233491e-01  2.7796891e-01  2.9566234e-01 -1.6812569e-01
 -5.5741054e-01 -1.6483348e-02 -2.4293737e-01 -4.5842607e-02
  4.6647485e-02  5.5738932e-01  1.9676497e-03 -7.3626889e-03
 -3.8457316e-01  4.6780738e-01  9.1540562e-03 -2.3180033e-01
  3.5629550e-01 -3.0026513e-01  4.5113844e-01  1.7072648e-01
 -2.8894678e-01  6.2121172e-02 -1.8993936e-01 -9.8223388e-02
  2.4271946e-01  1.2728746e-01 -7.1009614e-02 -9.2338465e-02
  1.5230374e-01  2.4899845e-01 -1.7543328e-01  8.2099274e-02
 -1.5266865e-01 -6.6597477e-02 -2.1794136e-01 -4.8963033e-02
 -1.8263882e-02  2.8120869e-01  1.7436291e-01  7.5858635e-01
  2.8531607e-02 -2.6733029e-01 -7.0961195e-01  7.9742931e-03
 -2.3128318e-02 -6.4376378e-01 -5.7583439e-01 -2.1754137e-01
 -1.3716704e-01 -2.3638194e-02  2.0786472e-01 -1.6847879e-01
 -8.9931879e-03 -2.4209489e-01  5.8387447e-02  2.3199101e-01
 -1.8911776e-01  3.0887327e-01  8.9017130e-02 -2.7528104e-01
  2.1195567e-01  1.5628964e-01  1.4287248e-01 -2.1467125e-01
  4.8996028e-01 -3.2011211e-02 -5.2048451e-01 -1.9626184e-01
 -5.7168806e-01  1.1026903e-01 -4.7341045e-02 -4.4059385e-02
 -9.2191756e-02  5.9221056e-04 -3.3218685e-01 -1.6703604e-01
  2.7131605e-01  2.1728919e-01  7.5621367e-02  3.3892530e-01
 -4.9175650e-01  3.9998397e-01 -1.7616749e-01 -1.8877859e-01
  3.9684433e-01  3.8221970e-01 -2.1146988e-02  7.8612790e-02]
```

1. text
2. nlp = spacy.load("en_core_web_sm")
3. convert this text into doc --> doc = nlp(text)

```
# NER
print(doc)
for ent in doc.ents:
    print(ent.text ,ent.start_char,ent.end_char, ent.label_)
```

```
My name is munira fatima and I am living in India along with my
parents and working for Microsoft
munira fatima 11 24 PERSON
India 44 49 GPE
Microsoft 88 97 ORG
```

Sentiment Analysis

```
review = "This product is very good"
```

```
!pip install textblob
```

```
Requirement already satisfied: textblob in c:\users\dell\anaconda3\
lib\site-packages (0.17.1)
Requirement already satisfied: nltk>=3.1 in c:\users\dell\anaconda3\
lib\site-packages (from textblob) (3.6.1)
Requirement already satisfied: regex in c:\users\dell\anaconda3\lib\
site-packages (from nltk>=3.1->textblob) (2021.4.4)
Requirement already satisfied: click in c:\users\dell\anaconda3\lib\
site-packages (from nltk>=3.1->textblob) (7.1.2)
Requirement already satisfied: joblib in c:\users\dell\anaconda3\lib\
site-packages (from nltk>=3.1->textblob) (1.3.2)
Requirement already satisfied: tqdm in c:\users\dell\anaconda3\lib\
site-packages (from nltk>=3.1->textblob) (4.66.1)
Requirement already satisfied: colorama in c:\users\dell\anaconda3\
lib\site-packages (from tqdm->nltk>=3.1->textblob) (0.4.6)
```

```
from textblob import TextBlob
```

```
review = "This product is worst"
doc = nlp(review)
```

```
filter_review = [token.text for token in doc if not token.is_stop]
print(filter_review)
```

```
['product', 'worst']
```

```
new_review = " ".join(filter_review)
print(new_review)
```

```
product worst
```

```
print(TextBlob(new_review).sentiment.polarity)
```

```
-1.0
```

```
review_sentiment = TextBlob(review).sentiment.polarity
```

```
if review_sentiment>0:
    print(f"{review}: Positive review")
else:
    print(f"{review}: Negative review")
```

```
This product is worst: Negative review
```

```
# Bag of words
# Tfidf

# used to convert your text into vectors

!pip install sklearn
```

```
Requirement already satisfied: sklearn in c:\users\dell\anaconda3\lib\
site-packages (0.0)
Requirement already satisfied: scikit-learn in c:\users\dell\
anaconda3\lib\site-packages (from sklearn) (1.3.2)
Requirement already satisfied: joblib>=1.1.1 in c:\users\dell\
anaconda3\lib\site-packages (from scikit-learn->sklearn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\dell\
anaconda3\lib\site-packages (from scikit-learn->sklearn) (2.1.0)
Requirement already satisfied: scipy>=1.5.0 in c:\users\dell\
anaconda3\lib\site-packages (from scikit-learn->sklearn) (1.10.1)
Requirement already satisfied: numpy<2.0,>=1.17.3 in c:\users\dell\
anaconda3\lib\site-packages (from scikit-learn->sklearn) (1.23.5)
```

```
import sklearn
from sklearn.feature_extraction.text import CountVectorizer # bag of
words

s1 = "This is a first document"
s2 = "This document is the second document"
# s3 = "And this is the third one"
# s4 = "Is this the first document"

corpus = [s1,s2]
print(corpus)
```

```
['This is a first document', 'This document is the second document']
```

```
vector = CountVectorizer()
x = vector.fit_transform(corpus)
print(x.toarray())
```

```
[[1 1 1 0 0 1]
 [2 0 1 1 1 1]]
```

```
vocab = []
print(s1)
for sentence in corpus:
    words_ls = sentence.split()
    for word in words_ls:
        if word not in vocab:
            vocab.append(word)
print(vocab)
```

```
This is a first document
['This', 'is', 'a', 'first', 'document', 'the', 'second']

unique_words = []
for sent in corpus:
    word_ls = sent.split()
    for word in word_ls:
        new_word = word.lower()
        if new_word not in unique_words:
            unique_words.append(new_word)
print(unique_words)

['this', 'is', 'a', 'first', 'document', 'the', 'second']

s1 = [1 1 1 1 1 0 0]

print(s2)

This document is the second document

s2 = [1 1 0 0 2 1 1]



from sklearn.feature_extraction.text import CountVectorizer
vector = CountVectorizer(ngram_range = (1,2))
x = vector.fit_transform(corpus)
print(x.toarray())

[[1 0 1 1 1 1 0 0 0 0 0 1 0 1]
 [2 1 0 0 1 0 1 1 1 1 1 1 1 0]]

from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer()
x = vectorizer.fit_transform(corpus)
print(x.toarray())

[[0.44832087 0.63009934 0.44832087 0.         0.         0.44832087]
 [0.63402146 0.         0.31701073 0.44554752 0.44554752 0.31701073]]
```