Extraktionsbaserad textsammanfattare med olika rankningsmått

Imports

In [1]:

```
# Imports --> TODO fixa ordning
# Web scrapping -->  module för att ladda ner artiklar
from newspaper import Article
import pandas as pd
from sklearn.preprocessing import StandardScaler
import numpy as np
import nltk
import ssl
import re  # används ej någonstans (tror jag) / björn
import regex as rex #


# Summarization length of original text
percentage = 0.15
n_sentences = 3

# Fixes some errors, found online at https://github.com/gunthercox/ChatterBot/issues/930#issuecomment-322111087
try:
    _create_unverified_https_context = ssl._create_unverified_context
except AttributeError:
    pass
else:
    ssl._create_default_https_context = _create_unverified_https_context
```

Input

Lägg till nya artiklar

- **Hämta komplett url**
- **Lägg till i python cellen under här**
- **Tilldela till variabeln text**
- **Tryck Restart och sen Run All**
- **Filerna hamnar i mappen "summarizations"**

In [3]:

```
text = "https://www.aftonbladet.se/nyheter/a/kE6ExL/sd-far-tunga-poster-i-utskotten"
#text = 'https://www.svt.se/nyheter/utrikes/stall-dina-fragor-om-kriget-till-svt-s-utrikesreportrar'
#text = "https://www.aftonbladet.se/nyheter/a/kE6j0a/uppgifter-viktigt-fynd-i-jakten-pa-mordarna-i-sodertalje"
#text = 'https://www.aftonbladet.se/nyheter/a/EQaJRo/blixtbygget-ska-radda-tyskland-i-vinter'
#text = 'https://www.aftonbladet.se/nyheter/a/3EEJGX/trump-vantas-kallas-att-vittna-om-6-januari'
```

In [4]:

```
# LÄnkar för Utvärdering Projekt --> Ran with percentage 0.3 & 0.15 for evaluation material
#text = 'https://www.aftonbladet.se/nyheter/a/EamE65/mannen-misstanks-ha-mordat-nancy-38--havdar-blackout'
#text = 'https://www.aftonbladet.se/nyheter/a/y66vWA/polisen-tackar-nej-till-hund-som-kunde-lost-mordet-pa-helena-andersson'
```

```
#text = 'https://www.aftonbladet.se/nojesbladet/melodifestivalen/a/wOJA0A/melodifestivalen-2022-appen-kraschar'
#text = 'https://www.svt.se/nyheter/inrikes/l-kd-och-m-ska-inga-i-regeringen-sd-far-stort-inflytande'
#text = 'https://www.svt.se/nyheter/inrikes/nord-stream-lackan-kan-ha-varit-medveten-attack'
```

In [5]:

```
article = Article(text, language='sv')
article.download()
article.parse()
text = article.text
homepage = article.meta_data['al']['ios']['app_name']
link = article.url
title = article.title

# Beroende på vilken hemsida nyheten kommer ifrån kan titeln och texten inehålla delar av sidan man egentligen inte bryr sig om
# T.ex. från aftonbladet är titeln med i texten och texten innehåller en mening som: "publicerad: 30 sep", man kan ta bort detta men
# det blir om vi får tid över.

#print('Title:' , article.title, '\n\nText: \n', text)
```

Preprocessing

Overview

Calculate number of sentences to keep

List 1 - sentences

- **Varje mening separat**

Dataframe - scores

columns are the score of each ranking measure

- **Baseline**
- **Headings**
- **TF/IDF-score**
- **NER**
- ~~**Class**~~ **//Om vi har tid för ML**

List 2 -> Cleaned for Stop Words

- **Varje mening separat**

#

Original Sentences List

In [6]:

```
# removes endlines:
from token import NEWLINE

org_sentences = text.replace('\n\n', '. ')
# creates some exceptions from above rule
org_sentences = org_sentences.replace('.. ', '. ')
```

```
org_sentences = org_sentences.replace(':. ', ': ')
org_sentences = org_sentences.split('. ')

# Detta borde egentligen tillhöra stopwordlistan
sentences_to_remove = []
for i, sentence in enumerate(org_sentences):
    if sentence == title:
        sentences_to_remove.append(sentence)
    if 'publicerad:' in sentence.lower():
        sentences_to_remove.append(sentence)
    if 'uppdaterad:' in sentence.lower():
        sentences_to_remove.append(sentence)

org_sentences = [sentence for sentence in org_sentences if sentence not in sentences_to_r
emove]

org_sentences[0:5]
```

Out[6]:

```
['Sverigedemokraterna får ordförandeposten i riksdagens justitie- och utrikesutskott',
 'Nu går ledarna för vänsterblocket till hård attack',
 '– Det är skrämmande, ganska chockartat, säger Socialdemokraternas gruppledare Lena Hall
engren till Aftonbladet',
 'Sverigedemokraterna , Moderaterna, Kristdemokraterna och Liberalerna har delat upp post
erna i utskotten och EU-nämnden',
 'Där tar Sverigedemokraterna flera viktiga poster']
```

Dataframe

In [7]:

```
index = range(org_sentences.__len__())
columns = ['Baseline', 'Headings', 'TF', 'NER']
scores = pd.DataFrame(index=index, columns=columns)
scores.fillna(0, inplace=True)
scores.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32 entries, 0 to 31
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Baseline  32 non-null     int64
 1   Headings  32 non-null     int64
 2   TF        32 non-null     int64
 3   NER       32 non-null     int64
dtypes: int64(4)
memory usage: 1.1 KB
```

Create spacy doc object

In [8]:

```
import spacy
# Credit to Explosion for sv_core_news_sm --> https://github.com/explosion
# "lemmatization accuracy 0.95"
# Create spacy nlp object
nlp = spacy.load("sv_core_news_sm") # nlp used by lemmatizer()
```

Lemmatizer

In [9]:

```
# den --> det
# noterar detta att det är en dålig lemmatiserare --> språkbanken stanza / lemmy / kth...
```

```
def lemmatizer(list_of_strings):
    lemmatized_sentences = []
    lemmatized_sentence = ''
    for i in range(len(list_of_strings)):
        sentence_to_lemmatize = nlp(list_of_strings[i])
        for token in sentence_to_lemmatize:
            lemma = token.lemma_
            lemmatized_sentence += lemma + ' '

        lemmatized_sentences.append(lemmatized_sentence)
        lemmatized_sentence = ''

    return lemmatized_sentences

# Created Lemmatized DS
lemmatized_org_sentences = lemmatizer(org_sentences)
print(lemmatized_org_sentences)
```

```
['Sverigedemokraterna få ordförandepost i riksdag justitie och utrikesutskott ', 'nu gå l
edarna för vänsterblock till hård attack ', '- den vara skrämmande , ganska chockarta , s
äga Socialdemokraterna gruppledare Lena Hallengren till Aftonbladet ', 'Sverigedemokrater
na , Moderat , Kristdemokraterna och Liberalerna ha dela upp post i utskott och EU-nämnde
n ', 'där ta Sverigedemokraterna flera viktig post ', 'bland annan tilldela parti ordföra
ndepost i arbetsmarknadsutskott , näringsutskott , justitieutskott samt utrikesutskottet
, enligt en pressmeddelande ', 'de erhåller även post som vice ordförande i civilutskott
, trafikutskott , försvarsutskotte samt skatteutskotte ', '- den som överraska jag mycket
, men som jag kunna se varför de vilja ha , vara ordförandepost i utrikesutskott ', 'den
vara en tecken på att de lycka i förhandling med M och KD , säga Aftonbladet My Rohwedder
i Aftonbladet tv ', '” få liten inflytande ” ', 'Nomineringarna väcka reaktion inom vänst
erblock ', '- den vara skrämmande , ganska chockarta ', 'Utrikesutskottet vara inte vilke
n som gärna , den skola representera Sverige riksdag utomlands och överhuvudtag i relatio
n till omvärld , så den vara en viktig position ', 'och här ha Moderat sålt sig ', 'den v
ara tyvärr en tecken på vart vi vara på väg , att för Ulf Kristersson del vara den viktig
att få makt och inflytande , man sälja sig billig , säga Lena Hallengren , gruppledare fö
r Socialdemokrater och fortsätta : - Utrikesutskottet vara otrolig betydelsefull med en o
mvärld som vara lite säga orolig ', 'att i en läge peka ut en sverigedemokrat , en parti
med rött i nazism , som den som skola företräd svensk folk i riksdag utrikes . ', 'på sam
ma sätt som jag nog vara lite skaka av denna vara den nog ingenting mot vad mången av en
land som ge vi säkerhetsgaranti vara , som undra vart Sverige vara på väg ', '” Skämmer u
t Sverige internationell ” ', 'flera politisk ledare protestera mot beslut på social medi
', 'Miljöpartiets språkröra Märta stenevi skriva på twitter : ” så en högerextrem parti s
om inte kunna välja mellan Putin och bid skola leda utrikesutskott och försvarsutskotte '
, 'Finnas ju ingen som kunna kunna gå fel i denna ” ', 'V-ledar Nooshie Dadgostar instämm
a : ” Återigen skämma en icke-tillträdd regering ut Sverige internationell ', 'en här gån
g genom att ge Sverigedemokraterna ordförandepost i utrikesutskotte ” , skriva hon på twi
tter ', 'Vänsterpartiets riksdagsledamot ali Esbati lyfta också SD-ledamöterna Martin kin
nun och Marku Wiechel besök i Syrien 2017 där de mötte Assad-regim ', '” för utrikesutsko
tt anta jag att den stå mellan Marcus Wiechel och Martin Kinnunen , här bägge på delegati
onsresa på egen initiativ och mot UD : s avråda , till Syrien 2017 ” , skriva han på twit
ter ', 'även Centerpartiets avgående partiledare Annie Lööf gå till attack efter beslut :
” så M , KD och L ge SD i uppdrag att leda utrikes- och försvarspolitik i riksdag ', 'i d
enna känslig säkerhetspolitisk läge , när Sverige skola bli medlem i Nato och vara ordför
ande i EU ', 'den vara olycklig och komma påverka Sverige anseende i värld ” ', '” få lit
en inflytande ” ', 'Utrikespolitiska institutets direktör Jakob Hallgren tro dock inte at
t utskotte komma att ha någon stor inverkan på en svensk utrikespolitik ', '- den här var
a en oerhöra viktig post som kanske vara mycket representativ än annan utskottspost , men
man måste komma ihåg att inom utrikespolitik ha vi en regering , en utrikesminister och e
n försvarsminist som ha mycket mycket verkställande makt , säga han och tillägga : - fråg
a vara hur stor inflytande ordförandefrågorna få om riksdagsparti förhandla fram överensk
ommelsa i förväg ', 'då komma utskotten att få liten inflytande , säga han . ']
```

Proper Nouns

In [10]:

```
# För bättre täckning på NER
def proper_nouns(list_of_strings):
    proper_nouns = set()
    for i in range(len(list_of_strings)):
        sentence_to_pos = nlp(list_of_strings[i])
```

```
        for token in sentence_to_pos:
            token_str = token.text
            if token.pos_ == "PROPN" and len(token_str) > 1:
                proper_nouns.add(token_str.strip())
    return proper_nouns

print(proper_nouns(org_sentences))
```

```
{'Kristerssons', 'SD', 'Wiechels', 'Martin', 'Aftonbladet', 'Socialdemokraternas', 'Biden
', 'EU', 'Sveriges', 'Putin', 'Nato', 'Annie', 'Markus', 'Kinnunen', 'Lena', 'KD', 'Esbat
i', 'UD', 'Sverige', 'Jakob', 'Stenevi', 'Wiechel', 'Rohwedder', 'Hallengren', 'Socialdem
okraterna', 'Syrien', 'Hallgren'}
```

Named Entities

In [11]:

```
def named_entity_recognition(list_of_strings):
    doc = nlp(' '.join(list_of_strings))
    # Convert tuple[Span] to str
    named_entities = doc.ents.__str__()
    # Remove string parenthesis
    named_entities = named_entities[1:len(named_entities) - 1]
    # Create list of strings
    named_entities = named_entities.split(',')

    named_entities_set = set()
    for entity in named_entities:
        named_entities_set.add(entity.strip())
    return named_entities_set
```

Stop Word Filtering

In [12]:

```
# Inspired by https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist

def get_swe_stop_words():
    swe_stop_words = set(stopwords.words('swedish'))
    # Not stop word cleaning
    swe_stop_words.update([',', '"', ':', '-', '–', '”'])
    return swe_stop_words

def stop_word_filtering(list_of_strings):
    word_tokens = word_tokenize(' '.join(list_of_strings))
    filtered_sentence_into_singletons = [w for w in word_tokens if not w.lower() in get_
swe_stop_words()]
    return filtered_sentence_into_singletons
```

Frequency Distribution

In [13]:

```
def frequency_distribution(list):
    fdist = FreqDist(word.lower() for word in word_tokenize(' '.join(list)))
    return fdist
```

In [14]:

```
filtered_words = stop_word_filtering(lemmatized_org_sentences)
fdist = frequency_distribution(filtered_words)
```

Ranking Measures

Baseline

(1/N --> n=ordning mening kommer i dvs. första meningen får N=1 -> 1/1, andra meningen får N=2 -> 1/2, osv.)

In [15]:

```
# Ranking metric 1 --> Baseline
for i, score in enumerate(scores['Baseline']) :
    scores['Baseline'][i] = 1/((i+1))
scores.describe()
```

Out[15]:

	Baseline	Headings	TF	NER
count	32.000000	32.0	32.0	32.0
mean	0.126828	0.0	0.0	0.0
std	0.188323	0.0	0.0	0.0
min	0.031250	0.0	0.0	0.0
25%	0.041250	0.0	0.0	0.0
50%	0.060662	0.0	0.0	0.0
75%	0.114583	0.0	0.0	0.0
max	1.000000	0.0	0.0	0.0

Headings

In [16]:

```
# Ranking metric 2 --> Headings
# Sets all 'Headings' scores to 0, mostly for testing so i can run this multiple times,
# but also to make sure nothing weird has happened earlier in the code.
scores['Headings'] = 0
for i, sentence in enumerate(org_sentences): # Lemmatizerad?
    for word in article.title.split(' '):
        if word in sentence:
            scores.at[i, 'Headings'] += 1
scores.describe()
```

Out[16]:

	Baseline	Headings	TF	NER
count	32.000000	32.000000	32.0	32.0
mean	0.126828	1.218750	0.0	0.0
std	0.188323	0.608243	0.0	0.0
min	0.031250	0.000000	0.0	0.0
25%	0.041250	1.000000	0.0	0.0
50%	0.060662	1.000000	0.0	0.0
75%	0.114583	1.000000	0.0	0.0
max	1.000000	3.000000	0.0	0.0

Sverigedemokraterna få ordförandepost i riksdag justitie och utrikesutskott [' få ', 'ordförandepost ', 'riksdag ', 'utrikesutskott ']

- **bara ord i lemma**
- **ord i lemma med efter letters**
- **ord i lemma med före letters --> x**
- **före + lemma + efter**

In [17]:

```
preprocessed_title = stop_word_filtering(lemmatizer(title.split(' ')))
p = rex.compile(r"\b\S*\L<words>\S*\b | \b\L<words>\b | \b\L<words>\S*\b", re.IGNORECASE
, words=preprocessed_title)
p.findall

for i, sentence in enumerate(lemmatized_org_sentences):
    #print(sentence)
    matches = p.findall(sentence)
    #print(matches)
    scores.at[i, "Headings"] = len(matches)

scores.describe()
```

Out[17]:

	Baseline	Headings	TF	NER
count	32.000000	32.000000	32.0	32.0
mean	0.126828	1.312500	0.0	0.0
std	0.188323	1.490562	0.0	0.0
min	0.031250	0.000000	0.0	0.0
25%	0.041250	0.000000	0.0	0.0
50%	0.060662	1.000000	0.0	0.0
75%	0.114583	2.000000	0.0	0.0
max	1.000000	5.000000	0.0	0.0

TERM FREQUENCY

In [18]:

```
# Borrowed https://www.askpython.com/python/examples/tf-idf-model-from-scratch (with adju
stments)
#Term Frequency
def termfreq(sentence, word):
    N = len(sentence)
    occurance = len([token for token in sentence if token == word])
    return occurance/N
```

In [19]:

```
# Borrowed https://www.askpython.com/python/examples/tf-idf-model-from-scratch (with adju
stments)
#Inverse Document Frequency
def inverse_doc_freq(word):
    try:
        word_occurance = fdist.get(word) + 1
    except:
        word_occurance = 1
    return np.log(len(lemmatized_org_sentences)/word_occurance)
```

In [20]:

```
# Borrowed https://www.askpython.com/python/examples/tf-idf-model-from-scratch (with adju
stments)
# TF*IDF-score
def tf_idf(sentence):
    score = 0
    #tf_idf_vec = np.zeros((len(filtered_words),))
    for word in sentence:
        tf = termfreq(sentence,word)
        idf = inverse_doc_freq(word)
```

```
        score += tf*idf
        #tf_idf_vec[index_dict[word]] = value
    return score

for i, sentence in enumerate(lemmatized_org_sentences):
    scores.at[i, 'TF'] = tf_idf(sentence)

#scores.describe()
#scores['TF'].idxmax()
#scores.loc[scores['TF'] == scores['TF'].median()]
```

TF*IDF-score

https://forketyfork.medium.com/latex-math-formulas-a-cheat-sheet-21e5eca70aae

In [21]:

```
# Ranking metric 3 --> TF*IDF

# Term Weights --> Calculate importance of single words in text/doc
# Binary term weights --> document specific
# TF*IDF term weights --> document-collection specific

# Assign weights to each dimension (attr/word) of each sentence (record/example)

# Term Frequency (TF-score) --> TFij == frequency of the jth term in in the ith doc

# Inverse Document Frequency
# idf-score of the jth term measures the uniqueness of the jth term in the collection of
documents
# IDFj = log(M / Nj)
#
# M = total num of docs in collection
# Nj is the number of documents that contain the jth term

# HIGH TF*IDF-score
# Word frequent in document && Occur in few documents of the collection
# LOW TF*IDF-score
# Not present in document || present in all documents of the collection
```

NER

In [22]:

```
scores['NER'] = 0
named_entities = named_entity_recognition(org_sentences)
proper_nouns = proper_nouns(org_sentences)
ner_unique = named_entities.union(proper_nouns)
print(ner_unique)
```

```
{'Martin Kinnunen', 'Kristerssons', 'SD', 'Wiechels', 'Martin', 'Moderaterna', 'Marcus Wi
echel', 'Markus Wiechels', 'UD:s', 'Aftonbladet', 'Socialdemokraternas', 'Biden', 'EU', '
Sveriges', 'Putin', 'Nato', 'Twitter', 'Annie', 'Skämmer', 'Markus', 'Kinnunen', 'Lena',
'Jakob Hallgren', 'KD', 'Esbati', 'UD', 'Sverige', 'Lena Hallengren', 'Miljöpartiets', 'J
akob', 'Ulf Kristerssons', 'Ali', 'Stenevi', 'Wiechel', 'Märta', 'Rohwedder', 'Aftonblade
ts My Rohwedder', 'Hallengren', 'Socialdemokraterna', 'Syrien', 'Hallgren'}
```

In [23]:

```
p = rex.compile(r"\L<words>", words=ner_unique)

for i, sentence in enumerate(org_sentences):
    matches = p.findall(sentence)
    scores.at[i, "NER"] = len(matches)

scores.describe()
scores.head()
```

Out[23]:

	Baseline	Headings	TF	NER
0	1.000000	4	17.542233	1
1	0.500000	0	11.884586	0
2	0.333333	0	26.721661	3
3	0.250000	2	24.534428	3
4	0.200000	1	11.882315	1

Combination Function

Standardize

In [24]:

```
# Standardize
scores_standardized = StandardScaler.fit_transform(self=StandardScaler(), X=scores)
scores_standardized = pd.DataFrame(scores_standardized, columns=columns)
scores_standardized['Baseline'] *= 1.5
scores_standardized['Headings'] *= 2.5
scores_standardized['TF'] *= 0.3
scores_standardized

# ÄNDRA COL NAME TF col --> TF*IDF TODO
# COUNTER WEIGHT:
# IF -->
# Baseline score low (e.g. below median)
# && term_frequency high (note: not TF*IDF) (e.g. above upper quartile)
# && NER high (e.g. above upper quartile)
# THEN:
# add_counterweight(sentence, counter_weight)
# counter_weight:
# final rankposition = i -->  ∑ -1 (-1 may be too big weight)
```

Out[24]:

	Baseline	Headings	TF	NER
0	7.066120	4.579654	-0.181553	-0.267900
1	3.019884	-2.236575	-0.275455	-0.839420
2	1.671139	-2.236575	-0.029198	0.875140
3	0.996766	1.171539	-0.065501	0.875140
4	0.592143	-0.532518	-0.275493	-0.267900
5	0.322393	6.283711	0.159330	-0.839420
6	0.129716	6.283711	0.009519	-0.839420
7	-0.014793	1.171539	0.008490	-0.839420
8	-0.127188	-2.236575	0.013009	0.875140
9	-0.217105	-0.532518	-0.326027	-0.839420
10	-0.290673	-2.236575	-0.273380	-0.839420
11	-0.351979	-2.236575	-0.260954	-0.839420
12	-0.403854	1.171539	0.197186	-0.267900
13	-0.448318	-2.236575	-0.322740	-0.267900
14	-0.486854	1.171539	0.743527	0.875140
15	-0.520572	-0.532518	0.150270	-0.839420
16	-0.550324	-2.236575	0.393143	-0.267900

	Baseline	Headings	TF	NER
17	-0.576770	-2.236575	-0.294190	0.303620
18	-0.600432	-2.236575	-0.206028	-0.839420
19	-0.621728	1.171539	0.179702	2.589699
20	-0.640996	-2.236575	-0.139686	-0.839420
21	-0.658512	-2.236575	-0.066469	-0.267900
22	-0.674505	1.171539	-0.033807	0.303620
23	-0.689165	1.171539	0.007320	2.589699
24	-0.702653	-0.532518	0.347194	2.018179
25	-0.715103	1.171539	0.188446	0.875140
26	-0.726631	-2.236575	-0.062986	0.875140
27	-0.737335	-2.236575	-0.200864	-0.267900
28	-0.747301	-0.532518	-0.326027	-0.839420
29	-0.756603	-0.532518	0.145077	-0.267900
30	-0.765304	4.579654	0.994731	-0.839420
31	-0.773462	1.171539	-0.196586	-0.839420

Calculate Summarization Length (number of sentences)

In [25]:

```
num_of_org_sentences = len(org_sentences)
summarization_num_sentences = round(num_of_org_sentences * percentage)

print("summarization: ", summarization_num_sentences, "\noriginal: ", num_of_org_sentenc
es)
```

```
summarization:  5
original:  32
```

Combine

- **Combine the scores into one overall score**
- **add weight and/or ML if time allows**

In [26]:

```
# Combination Function
# Här ligger ML om vi gör det

#final_score = scores_standardized.drop('TF', axis=1).sum(axis=1)
final_score = scores_standardized.sum(axis=1)
best_sentences = final_score.nlargest(summarization_num_sentences, keep='all').index.valu
es
print(best_sentences)

print(org_sentences[0])
```

```
[ 0  5  6 30 19]
Sverigedemokraterna får ordförandeposten i riksdagens justitie- och utrikesutskott
```

In [27]:

```
# Similarity heurstic
# Rubrik --> Hur mycket lika --> Ta bort redundans
# 5 fem väldigt lika --> heuristik välja bort något av det
```

Assemble output

- **Reassemble according to overall score ranking**
- **Output summarization**

In [28]:

```
# Percentage length of original text --> summarization
print("Percentage:\n")
print(title, "\n")
for i in best_sentences:
    print(org_sentences[i])

print("\n\n")

# N sentences
print("N sentences:\n")
for i in best_sentences[0:n_sentences]:
    print(org_sentences[i])
```

```
Percentage:

SD får tunga poster i utskotten

Sverigedemokraterna får ordförandeposten i riksdagens justitie- och utrikesutskott
Bland annat tilldelas partiet ordförandeposten i arbetsmarknadsutskottet, näringsutskotte
t, justitieutskottet samt utrikesutskottet, enligt ett pressmeddelande
De erhåller även posten som vice ordförande i civilutskottet, trafikutskottet, försvarsut
skottet samt skatteutskottet
– Det här är en oerhört viktig post som kanske är mer representativ än andra utskottspost
er, men man måste komma ihåg att inom utrikespolitiken har vi en regering, en utrikesmini
ster och en försvarsminister som har mycket mer verkställande makt, säger han och tillägg
er: – Frågan är hur stort inflytande ordförandefrågorna får om riksdagspartierna förhandl
ar fram överenskommelser i förväg
Miljöpartiets språkrör Märta Stenevi skriver på Twitter: ”Så ett högerextremt parti som i
nte kunnat välja mellan Putin och Biden ska leda utrikesutskottet och försvarsutskottet



N sentences:

Sverigedemokraterna får ordförandeposten i riksdagens justitie- och utrikesutskott
Bland annat tilldelas partiet ordförandeposten i arbetsmarknadsutskottet, näringsutskotte
t, justitieutskottet samt utrikesutskottet, enligt ett pressmeddelande
De erhåller även posten som vice ordförande i civilutskottet, trafikutskottet, försvarsut
skottet samt skatteutskottet
```

In [29]:

```
# Newspaper Summarization
article.nlp()
print("\nNewspaper3k: \n", article.summary)
```

```
Newspaper3k:
 SD får tunga poster i utskottenPublicerad: 30 september Uppdaterad: 30 septemberSveriged
emokraterna får ordförandeposten i riksdagens justitie- och utrikesutskott.
Utrikesutskottet är inte vilket som helst, det ska representera Sveriges riksdag utomland
s och överhuvudtaget i relationen till omvärlden, så det är en viktig position.
Den här gången genom att ge Sverigedemokraterna ordförandeposten i utrikesutskottet”, skr
iver hon på Twitter.
I detta känsliga säkerhetspolitiska läge, när Sverige ska bli medlem i Nato och vara ordf
örande i EU.
”Får mindre inflytande”Utrikespolitiska institutets direktör Jakob Hallgren tror dock int
e att utskottet kommer att ha någon större inverkan på den svenska utrikespolitiken.
```

In [30]:

```
summarization = link + '\n\n' + 'Title: ' + '\n' + title + '\n\n' + 'Compression ratio:
' + str(len(best_sentences)/len(org_sentences)) + '\n\n' + 'Summarization:' + '\n'
for sentence in best_sentences:
    summarization += org_sentences[sentence] + '\n'
summarization += '\n\n' + 'Newspaper3k:' + '\n' + 'Title: ' + '\n' + title + "\n\n" + ar
```

```
ticle.summary
```

In [31]:

```
import os
FILEPATH = f'./summarizations/{homepage}/'
filename = f"{FILEPATH}{title}.txt"
os.makedirs(os.path.dirname(filename), exist_ok=True)

with open(filename, 'w') as f:
    f.write(summarization)
```