KED 2023 Miniprojekt

Textanalyse: Berichterstattung über Kryptowährungen

Nick Rölli, Cedric Keiser, Nils Wolf, Valentin Casanova

Übersicht

Thema: Häufigkeit und Wahrnehmung von **Artikeln über Kryptowährungen** in den letzten Jahren

Daten

Analysen:

- Publikationsverhalten
- Häufigkeit von Wörtern

Daten

Stammen von Swissdox

Keywords: “Kryptowährung”, “Bitcoin”

Zeitungen: NZZ, TA, SRF, WeWo

Zeitspanne: 2008 - May 2023

Publikationsverhalten

```python
#import
import pandas as pd
from bs4 import BeautifulSoup
import plotnine as pn

#Pfad
fname = "dataset_provisional.tsv"

# Dataset laden
df = pd.read_csv(fname, sep="\t")
```

```python
# Irrelevante Spalten eliminieren
columns_to_drop = [
    "id",
    "regional",
    "doctype",
    "doctype_description",
    "dateline",
    "language",
    "subhead",
    "content_id",
    "medium_name",
]
# Löschen und den Rest in Subset speichern
df_sub = df.drop(columns=columns_to_drop)
```

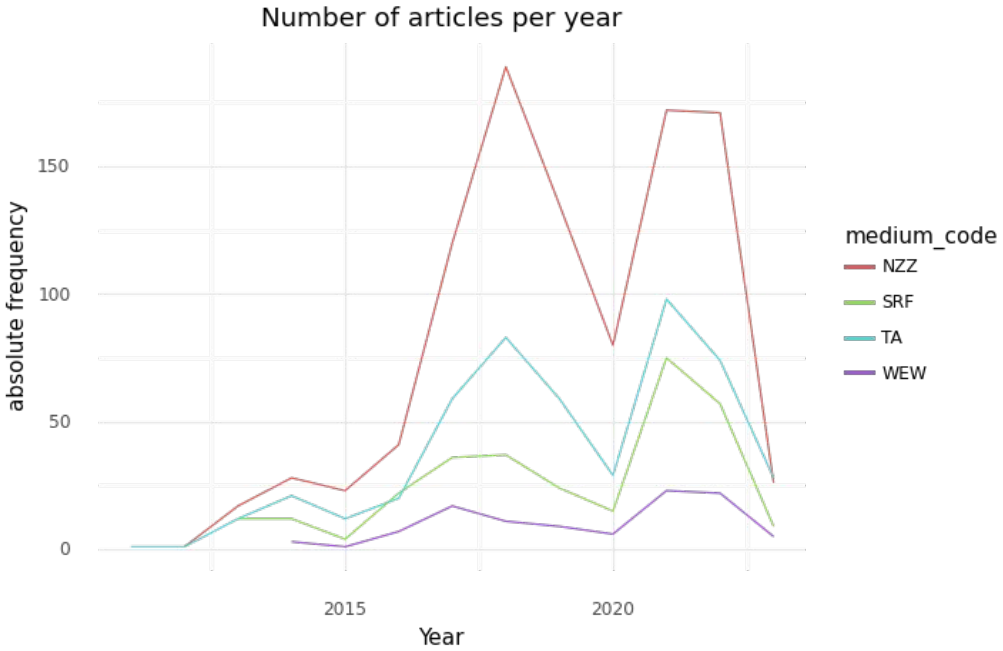
```python
# Tags, Autoren etc. löschen
def remove_html_tags(text):
    # create beautiful soup object for easy clean-up
    soup = BeautifulSoup(text)

    try:
        # remove the authors name in the tag, e.g. <au>AUTHORNAME</au>
        soup.au.decompose()
    except AttributeError:
        # in some of the articles, there is no <au>AUTHORNAME</au>
        # thus, we want to catch the error and just by-pass the raised error
        pass

    # strip the remaining tags and join the strings by newline
    text = soup.get_text(separator="\n", strip=True)

    return text


# Call the function remove_html_tags for each cell in the column `content`
df_sub["content"] = df_sub["content"].apply(lambda x: remove_html_tags(x))
```

```python
# Parse the `pubtime` as datetype (accounting for timezones)
df_sub["pubtime"] = pd.to_datetime(df_sub["pubtime"], utc=True)

# Create new columns with publication year and month
df_sub["year"] = df_sub["pubtime"].dt.year
df_sub["month"] = df_sub["pubtime"].dt.month

# Group data by year, month, and newspaper and count the respective articles
docs_per_month = (
    df_sub.groupby([pd.Grouper(key="pubtime", freq="M"), "medium_code"])
    .agg({"content": "count"})
    .reset_index()
    .rename(columns={"content": "count"})
)
docs_per_month["year"] = docs_per_month["pubtime"].dt.year
docs_per_month["month"] = docs_per_month["pubtime"].dt.month
docs_per_month.drop("pubtime", axis=1, inplace=True)

docs_per_month
```

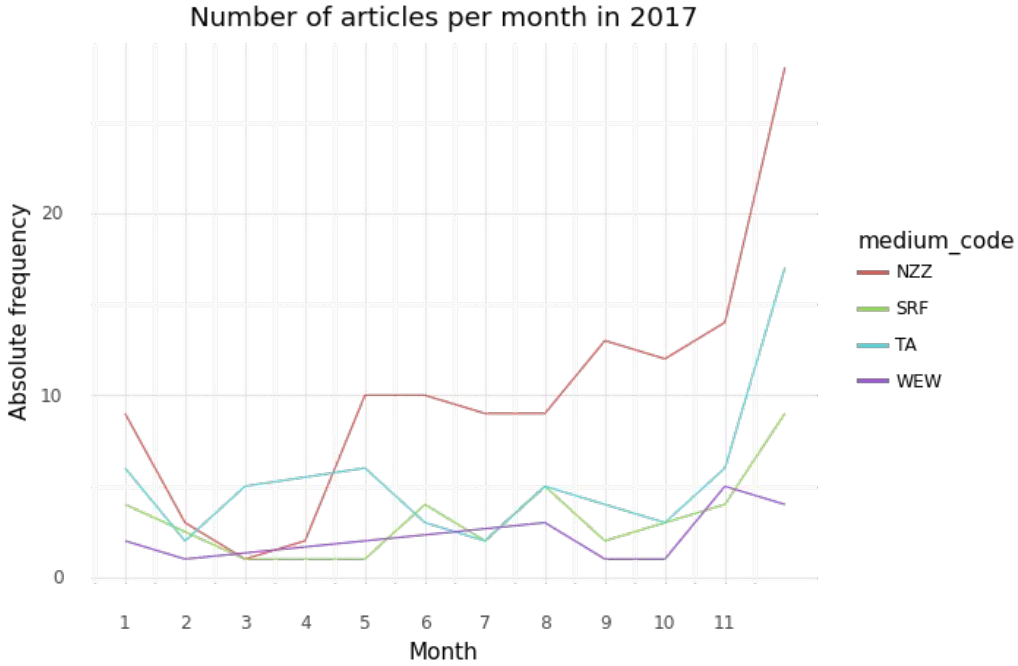
```python
# Visualization number of articles per year and newspaper
(
    pn.ggplot(docs_per_year, pn.aes(x="year", y="count", color="medium_code"))
    + pn.geom_line()
    + pn.labs(title="Number of articles per year", x="Year", y="absolute frequency")
    + pn.theme_minimal()
)
```

```python
# Visualization number of articles per year and newspaper

selected_year = 2021
(
    pn.ggplot(docs_per_month[docs_per_month["year"] == selected_year], pn.aes(x="month", y="count", color="medium_code"))
    + pn.geom_line()
    + pn.labs(title=f"Number of articles per month in {selected_year}", x="Month", y="Absolute frequency")
    + pn.scale_x_continuous(breaks=range(1, 12))  # Set the x-axis tick values from 1 to 12
    + pn.theme_minimal()
)
```

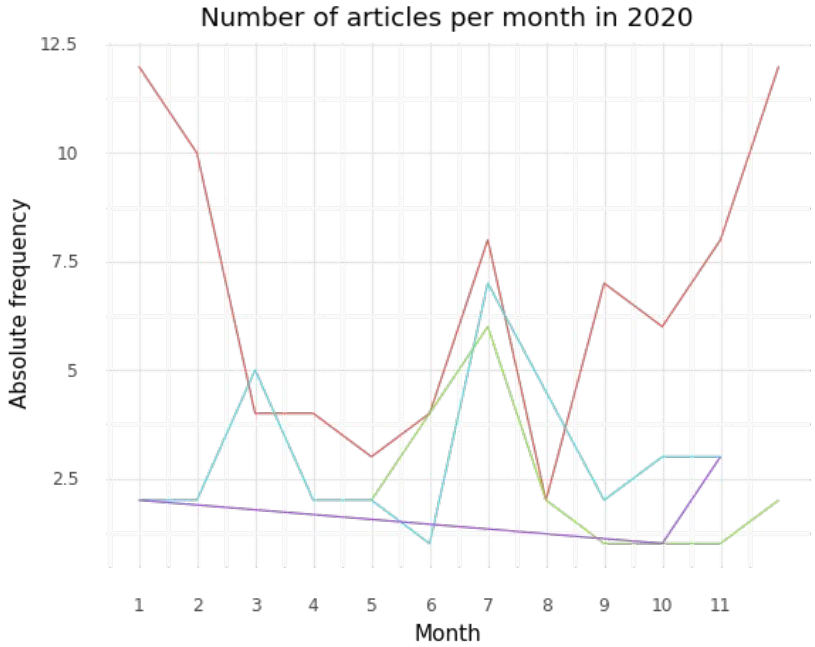
Resultate - Gesamt



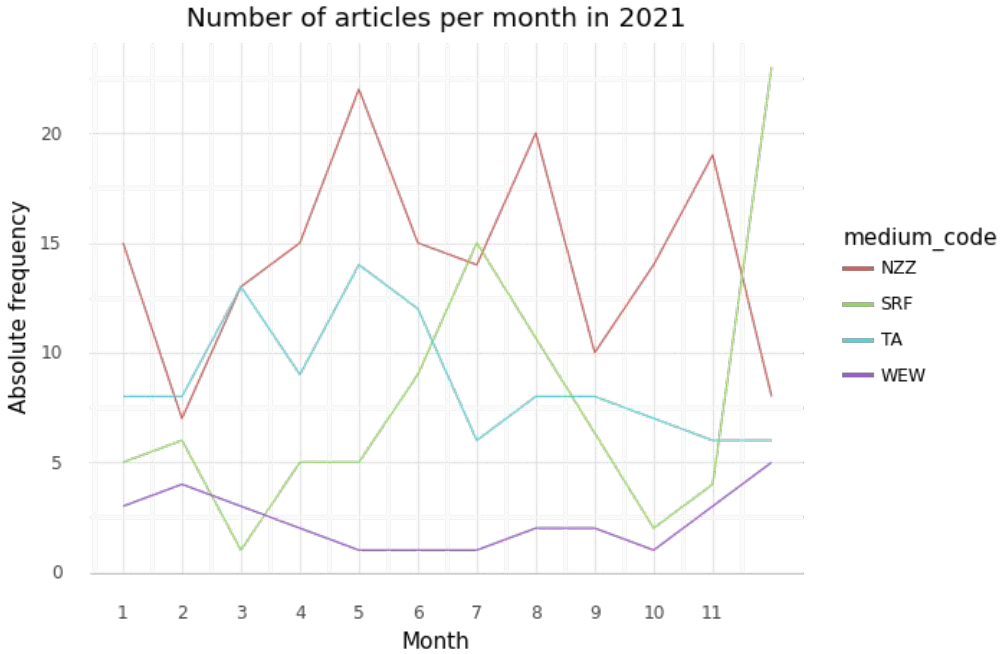
Resultate - 2017



Resultate - 2020



Resultate - 2021



Häufigkeit von Wörtern

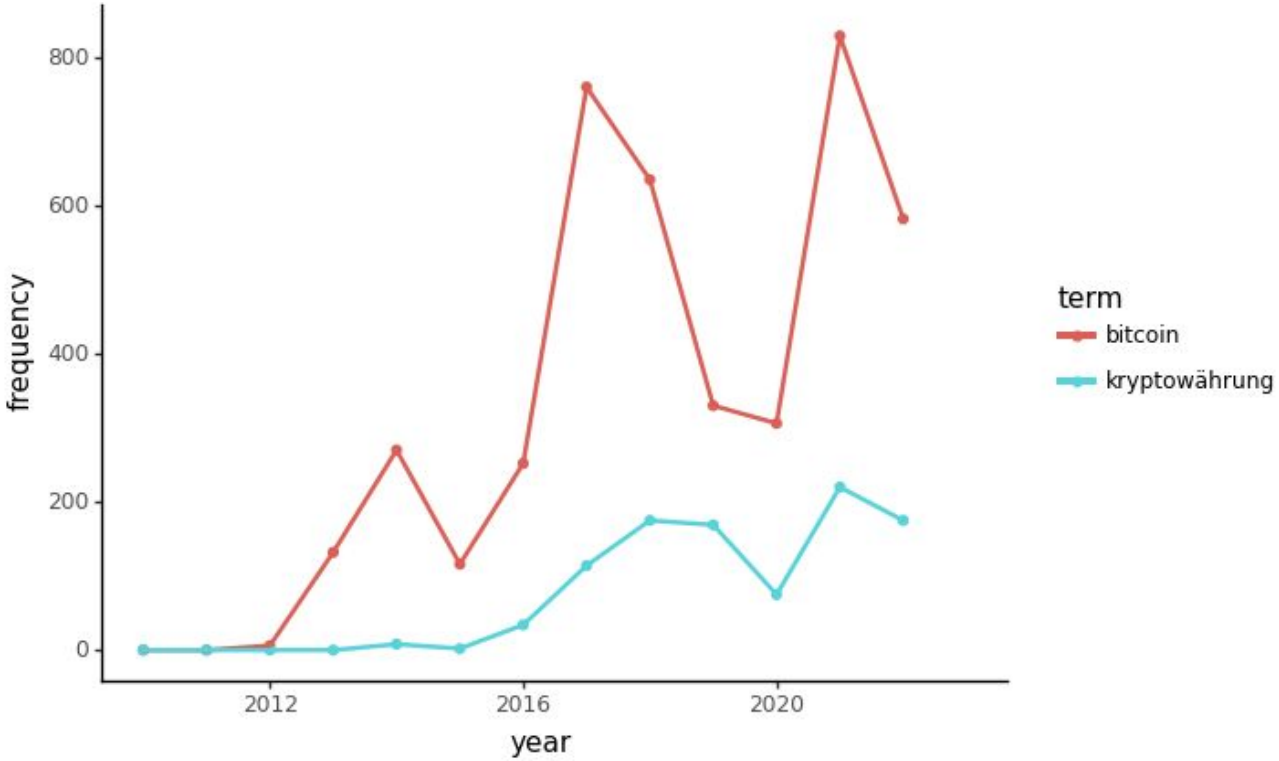
```python
#Import
import textacy
import spacy

def get_texts_from_csv(fname, text_column):
    df = pd.read_csv(fname, sep="\t")


    # keep only documents that have text
    filtered_df = df_sub[df_sub[text_column].notnull()]

    # iterate over rows in dataframe
    for idx, row in filtered_df.iterrows():

        # get text and join lines (remove hard line-breaks)
        text = row[text_column].replace("\n", " ")

        # use all columns as metadata, except the column with the actual text
        metadata = row.to_dict()
        del metadata[text_column]

        yield (text, metadata)

f_csv = "dataset_provisional.tsv"
texts = get_texts_from_csv(f_csv, text_column="content")

#Load german language model and create corpus
nlp = spacy.load('de_core_news_sm')
corpus_speeches = textacy.Corpus(nlp, data=texts)
```

✓ 6m 3.5s

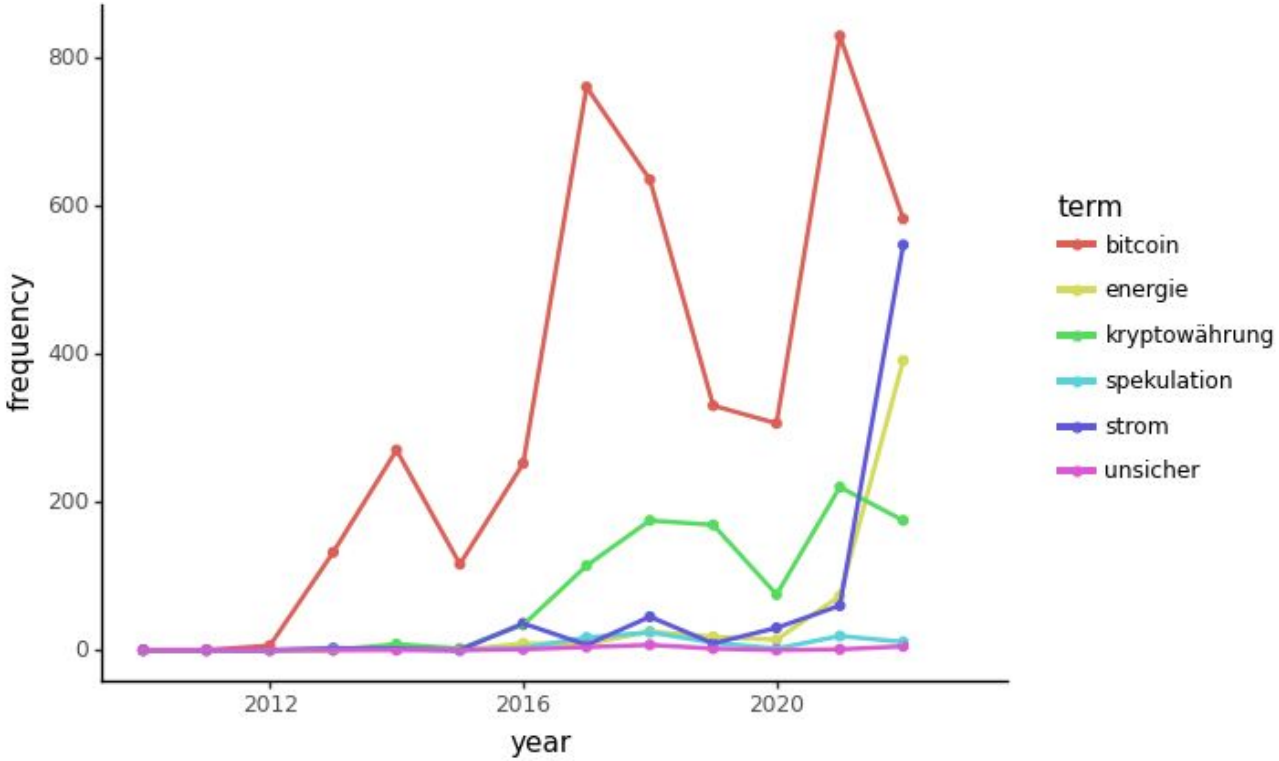
```python
# define what groups are formed and what terms should be included
# here, groups by year and words are lowercased (incl. stop words)
tokenized_docs, groups = textacy.io.unzip(
    (
        textacy.extract.utils.terms_to_strings(
            textacy.extract.words(doc, filter_stops=False), by="lower"
        ),
        doc._.meta["year"],
    )
    for doc in corpus_speeches
)

# define how to count
vectorizer = textacy.representations.vectorizers.GroupVectorizer(
    tf_type="linear",  # absolute term frequency
    vocabulary_grps=range(2010, 2023),
)  # limit to years from 2010 to 2023

# create group-term-matrix with with frequency counts
grp_term_matrix = vectorizer.fit_transform(tokenized_docs, groups)

# create dataframe from matrix
df_terms = pd.DataFrame.sparse.from_spmatrix(
    grp_term_matrix, index=vectorizer.grps_list, columns=vectorizer.terms_list
)
df_terms["year"] = df_terms.index

# change shape of dataframe
df_tidy = df_terms.melt(id_vars="year", var_name="term", value_name="frequency")
df_tidy
```

```python
# filter the dataset for the following terms
terms = ["bitcoin", "kryptowährung"]
df_terms = df_tidy[df_tidy["term"].isin(terms)]

# plot the relative frequency for the terms above
(
    pn.ggplot(df_terms, pn.aes(x="year", y="frequency", color="term"))
    + pn.geom_point()  # show individual points
    + pn.stat_smooth(
        method="lowess", span=0.15, se=False
    )  # overlay points with a smoothed line
    + pn.scale_x_continuous(limits=(2010, 2023))
    + pn.theme_classic()
)  # make the plot look nicer
```

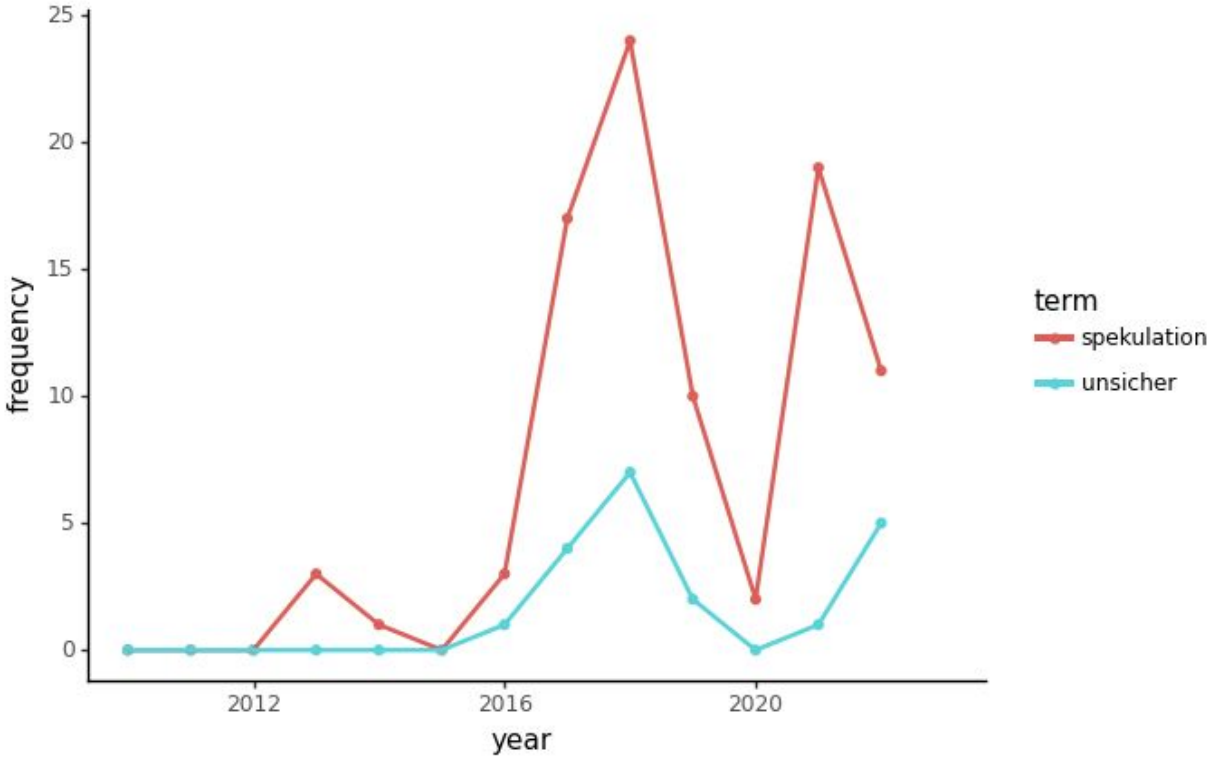
Resultate


800
600
400
200
0
frequency
2012
2016
2020
year
term
bitcoin
kryptowährung

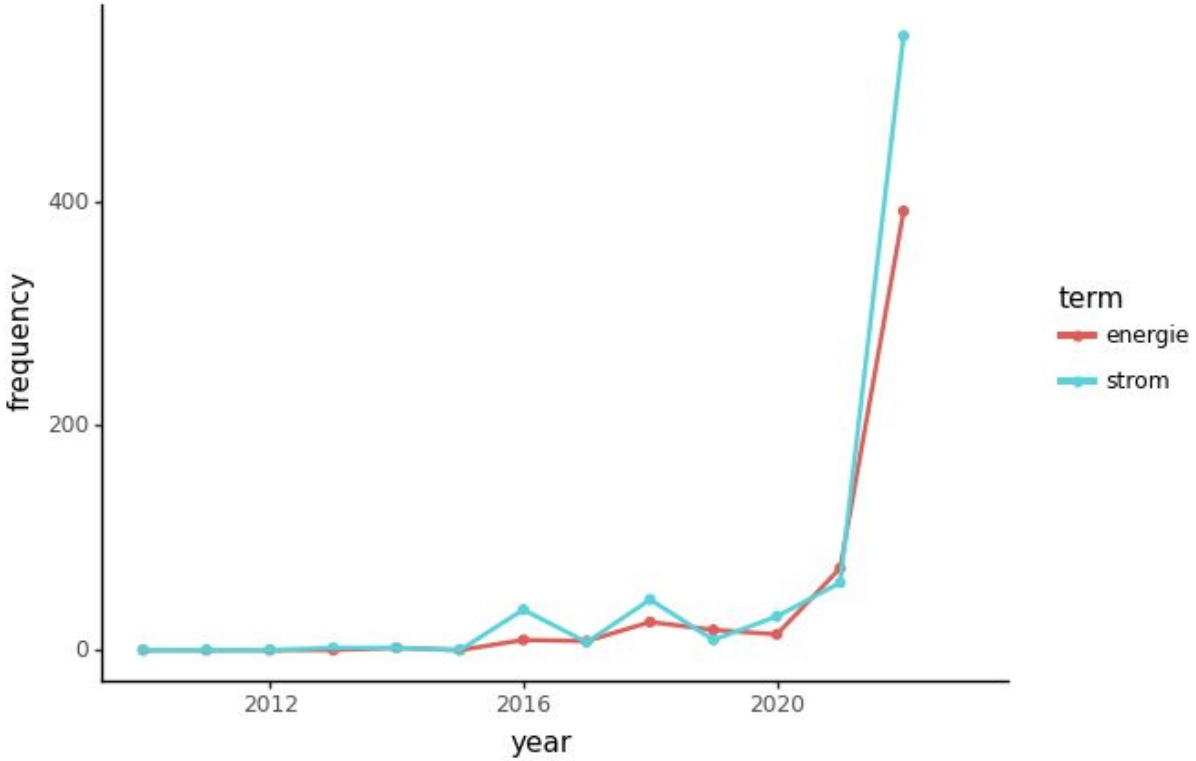
Resultate


800
600
400
200
0
frequency
2012
2016
2020
year
term
bitcoin
energie
kryptowährung
spekulation
strom
unsicher

Resultate


25
20
15
10
5
0
frequency
2012
2016
2020
year
term
spekulation
unsicher

Resultate



Fazit

- Berichterstattung korreliert stark mit den Preisschwankungen
- Zeitungen verhalten sich ähnlich, aber unterschiedlich gross bzw. anderer Fokus der Medien

- Thema Spekulation und Unsicherheit scheint mit Preisschwankungen zu korrelieren
- Energie-Thematik war lange nicht präsent, womöglich durch Ukraine Krieg verstärkt

Fragen?