12.01.22

OPENCAMPUS.sh

Natural Language Processing with Transformers

TRANSFORMER ARCHITECTURE AND DATA UTILITIES

- Quiz
- Transformer Architecture
- Data Visualization
- Importing Data
- Saving Models

QUIZ

https://forms.office.com/r/CyNs7LQG0A

WHY EMBEDDINGS AND THE DOT PRODUCT?

The one-hot encoding technique has two main drawbacks:

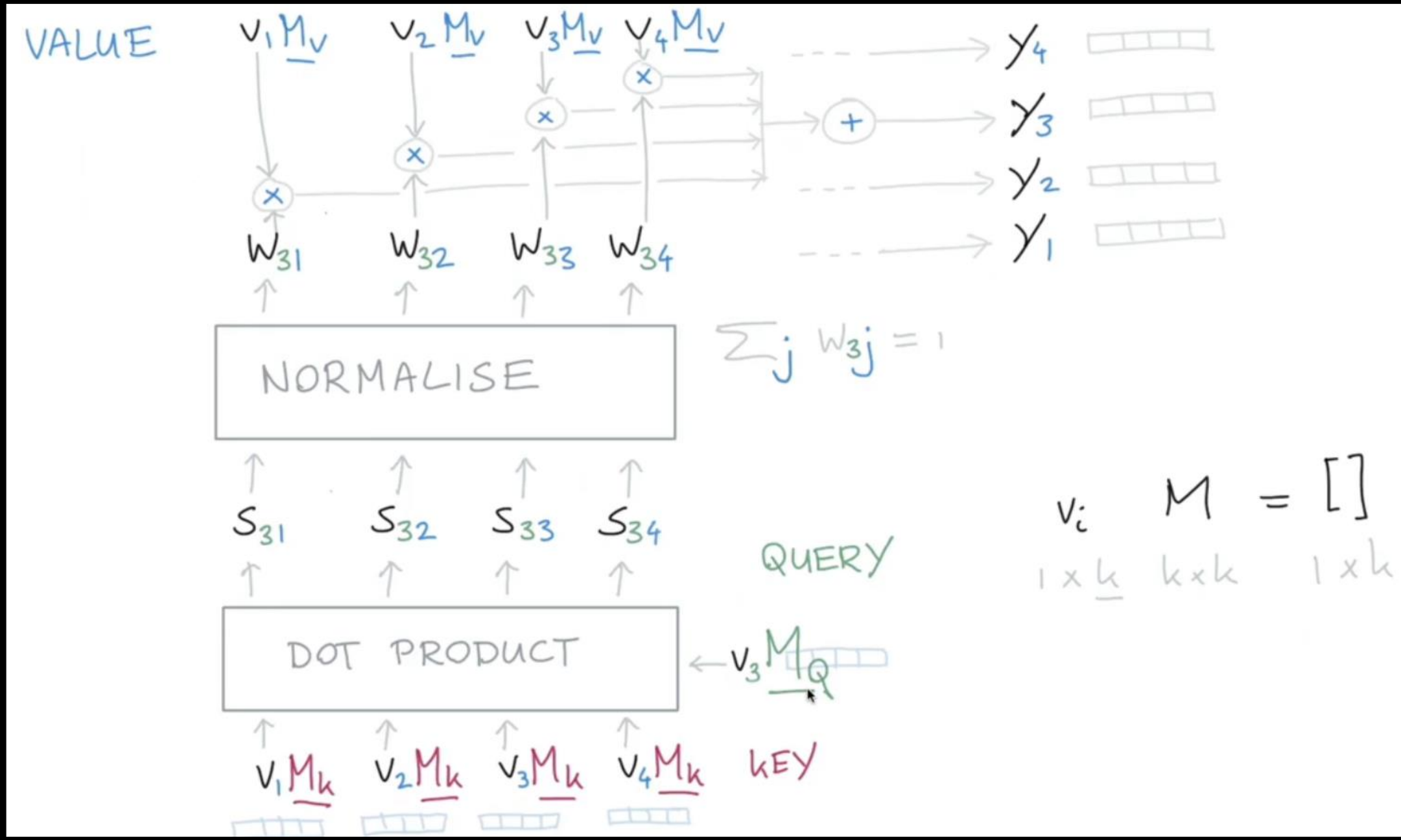
1. For high-cardinality variables — those with many unique categories — the dimensionality of the transformed vector becomes unmanageable.

2. The mapping is completely uninformed: “similar” categories are not placed closer to each other in embedding space.

DOT PRODUCT SIMILARITY

```
# One Hot Encoding Categoricals

books = ["War and Peace", "Anna Karenina",
          "The Hitchhiker's Guide to the Galaxy"]

books_encoded = [[1, 0, 0],
                 [0, 1, 0],
                 [0, 0, 1]]

Similarity (dot product) between First and Second = 0
Similarity (dot product) between Second and Third = 0
Similarity (dot product) between First and Third = 0
```

```
# Idealized Representation of Embedding

books = ["War and Peace", "Anna Karenina",
          "The Hitchhiker's Guide to the Galaxy"]

books_encoded_ideal = [[0.53,  0.85],
                       [0.60,  0.80],
                       [-0.78, -0.62]]

Similarity (dot product) between First and Second = 0.99
Similarity (dot product) between Second and Third = -0.94
Similarity (dot product) between First and Third = -0.97
```


VALUE
$v_1 M_V$
$v_2 M_V$
$v_3 M_V$
$v_4 M_V$
y_4
y_3
y_2
y_1
w_{31}
w_{32}
w_{33}
w_{34}
NORMALISE
$\sum_j w_{3j} = 1$
s_{31}
s_{32}
s_{33}
s_{34}
QUERY
DOT PRODUCT
$v_3 M_Q$
$v_1 M_k$
$v_2 M_k$
$v_3 M_k$
$v_4 M_k$
KEY
v_i $M = []$
1 x k k x k 1 x k

POSITIONAL ENCODING

Important Criteria:

- **It should output a unique encoding for each time-step (word's position in a sentence)**
- **Distance between any two time-steps should be consistent across sentences with different lengths.**
- **Our model should generalize to longer sentences without any efforts. Its values should be bounded.**
- **It must be deterministic.**

Amirhossein Kazemnejad. (2019, September 20). *Transformer Architecture: The Positional Encoding* [Blog]. Amirhossein Kazemnejad's Blog. https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

POSITIONAL ENCODING FUNCTION

$$\overrightarrow{p_t}^{(i)} = f(t)^{(i)} := \begin{cases} \sin(\omega_k . t), & \text{if } i = 2k \\ \cos(\omega_k . t), & \text{if } i = 2k+1 \end{cases}$$

where $$\omega_k = \frac{1}{10000^{2k/d}}$$

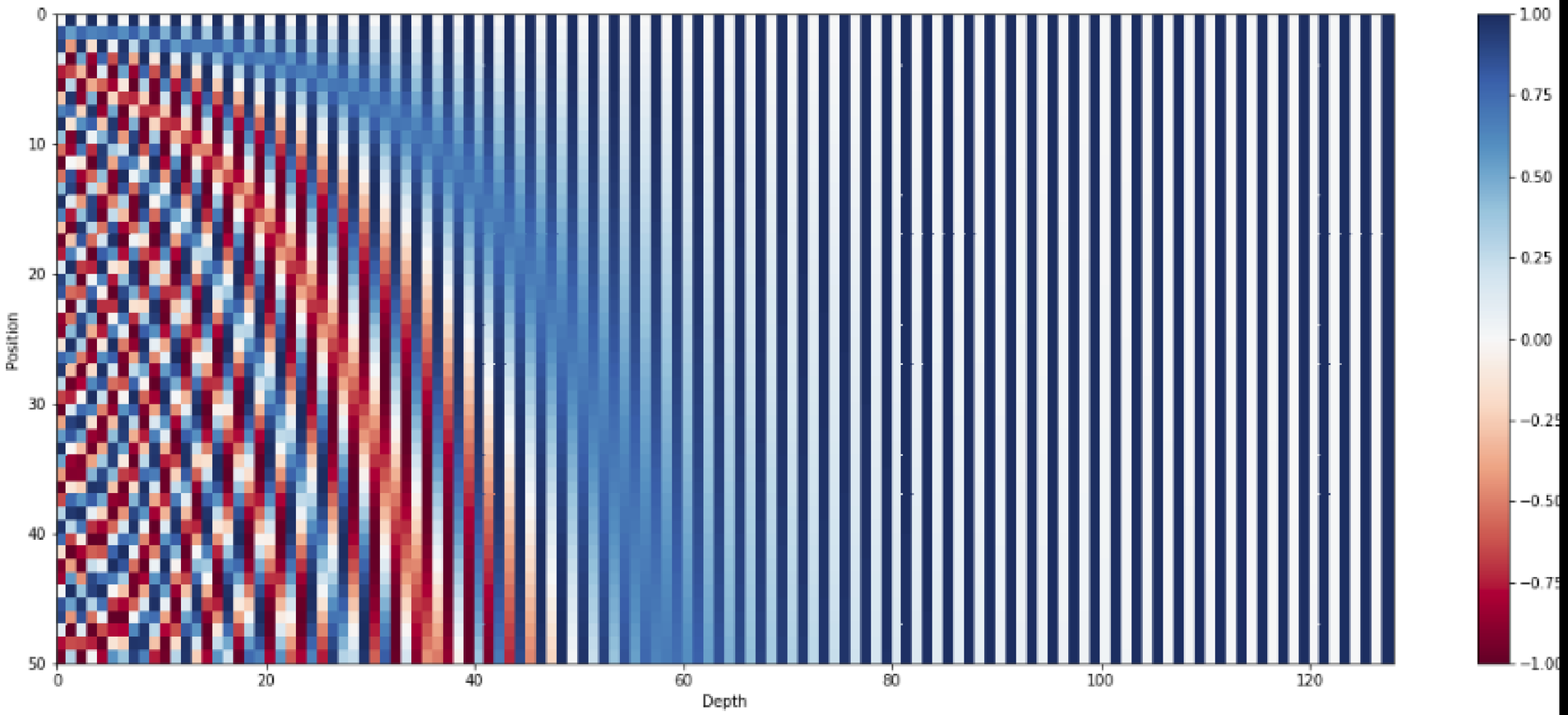
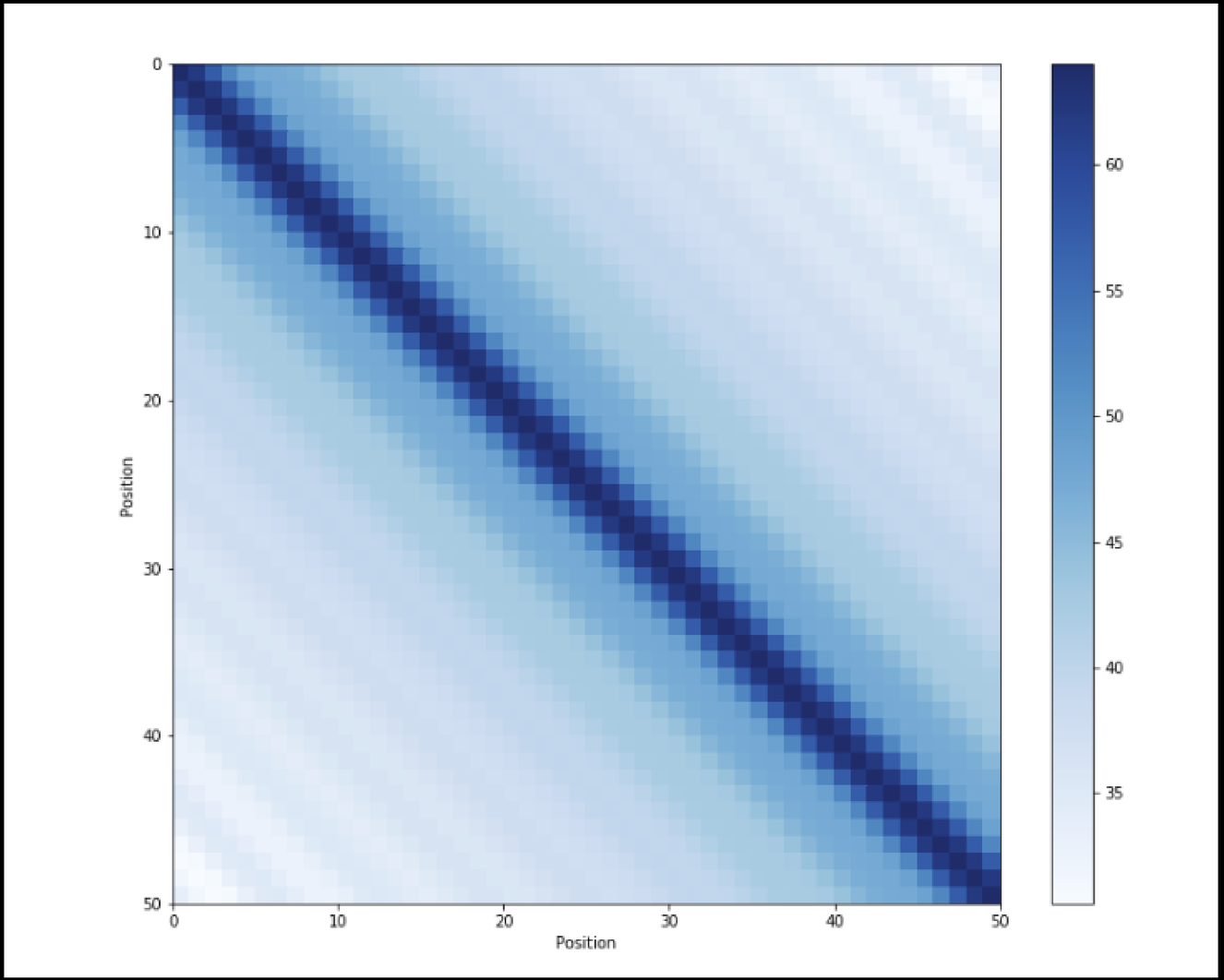


Figure 2 – The 128-dimensional positonal encoding for a sentence with the maximum lenght of 50. Each row represents the embedding vector $\overrightarrow{p_t}$

RELATIVE POSITIONING

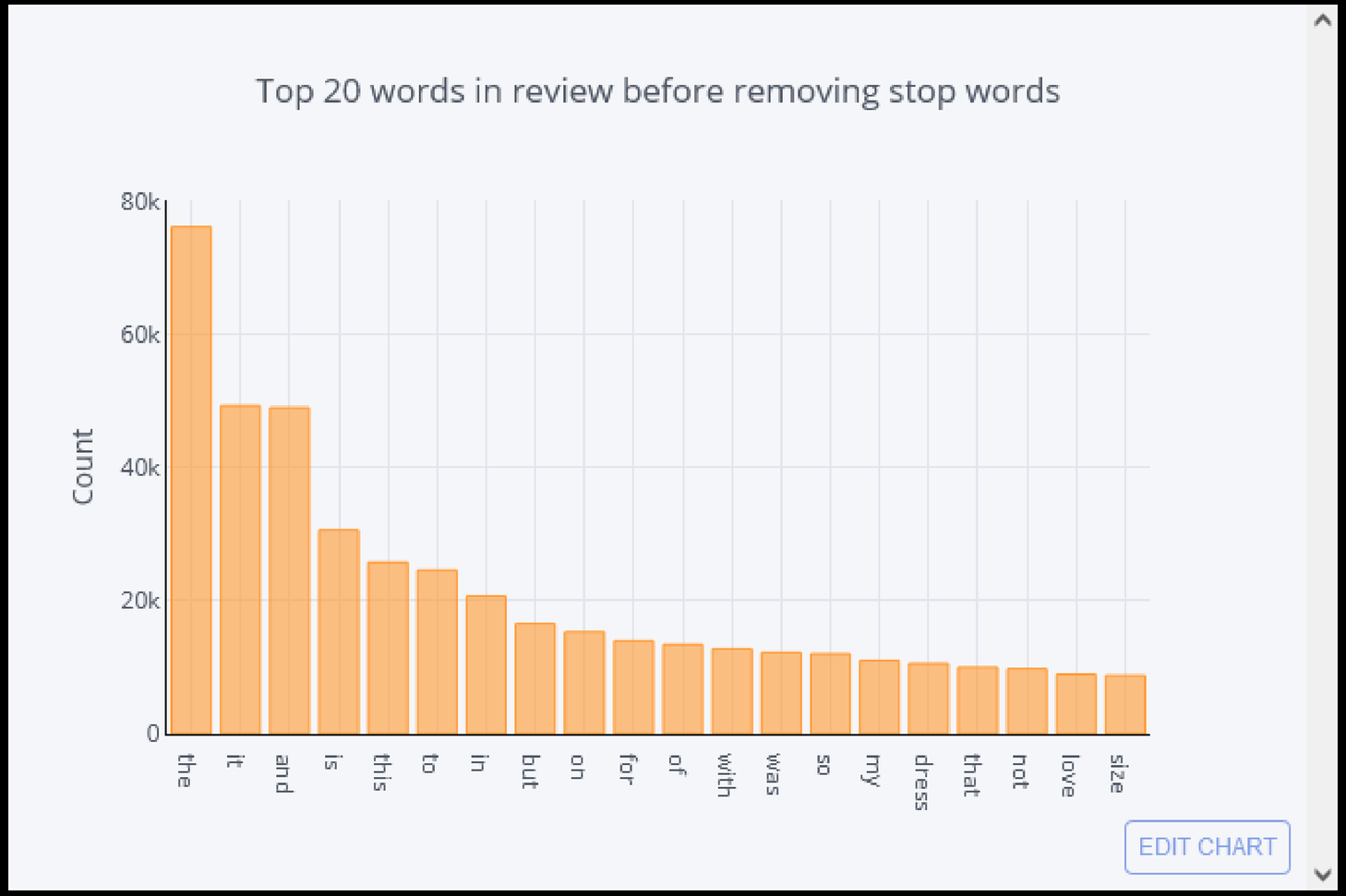


DATA VISUALIZATION

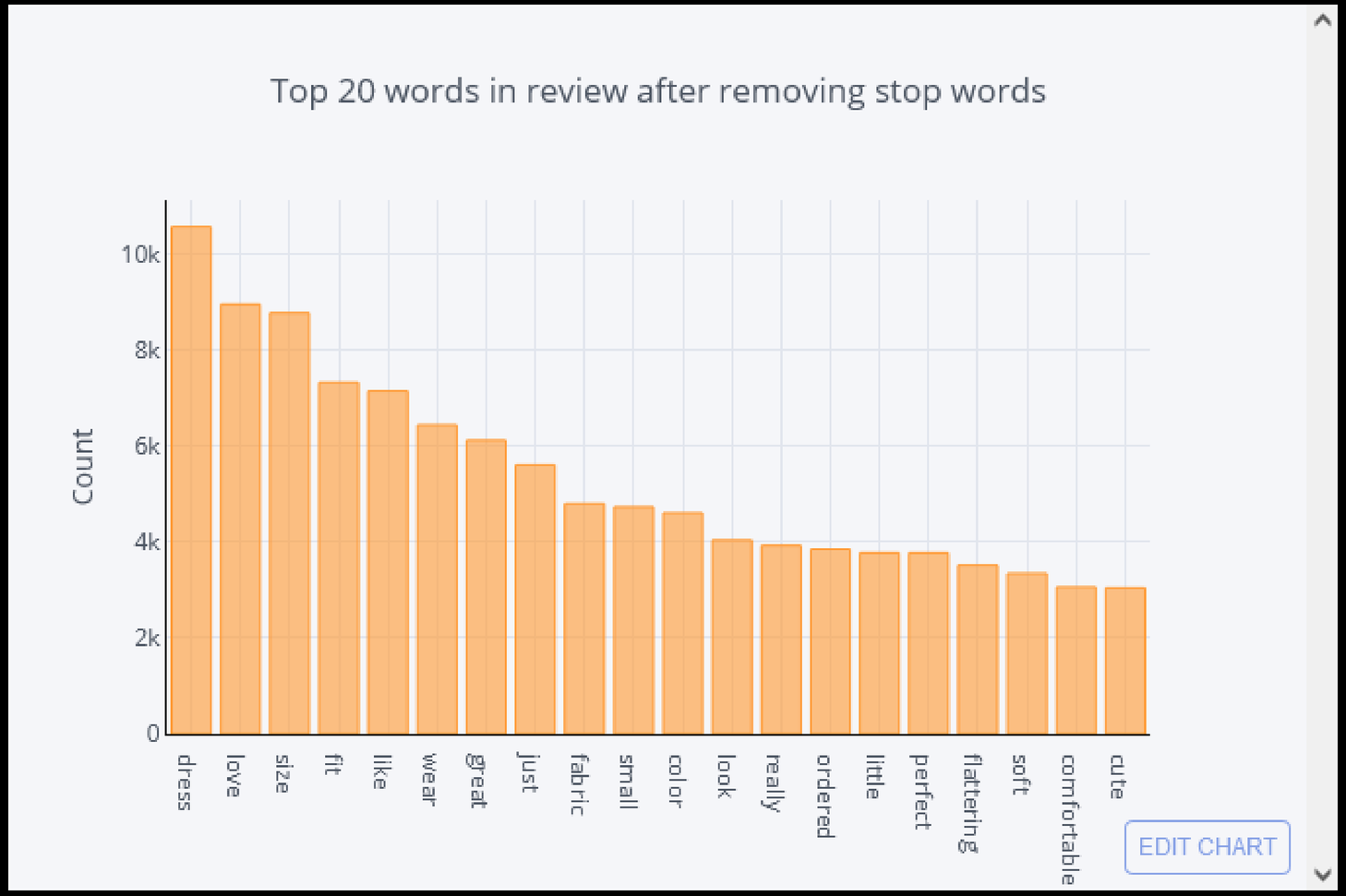
- Li, S. (2019, April 27). *A Complete Exploratory Data Analysis and Visualization for Text Data*. Medium. https://towardsdatascience.com/a-complete-exploratory-data-analysis-and-visualization-for-text-data-29fb1b96fb6a

- Example data using E-commerce reviews on cloth

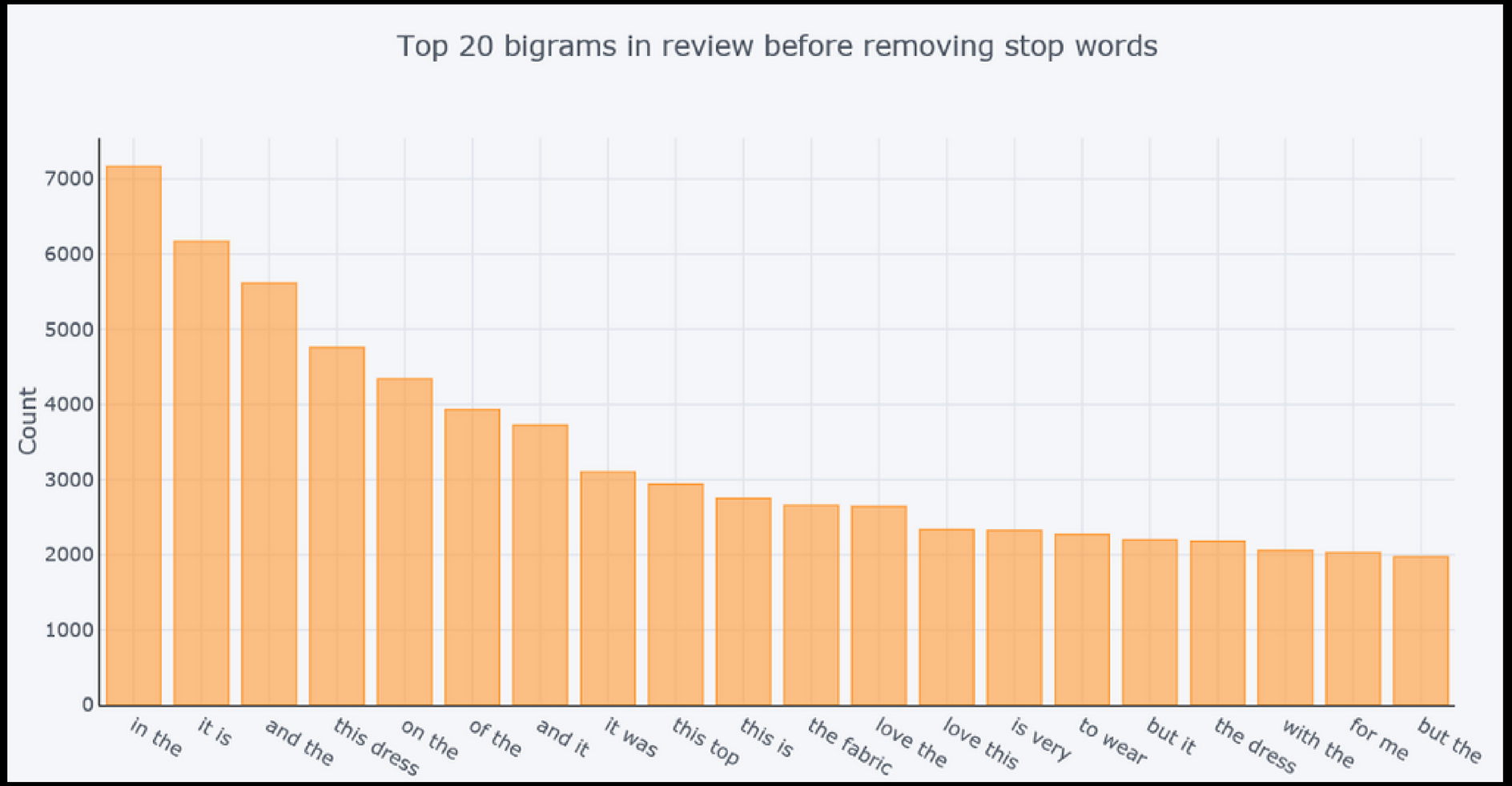
UNIGRAMS BEFORE REMOVING STOPWORDS



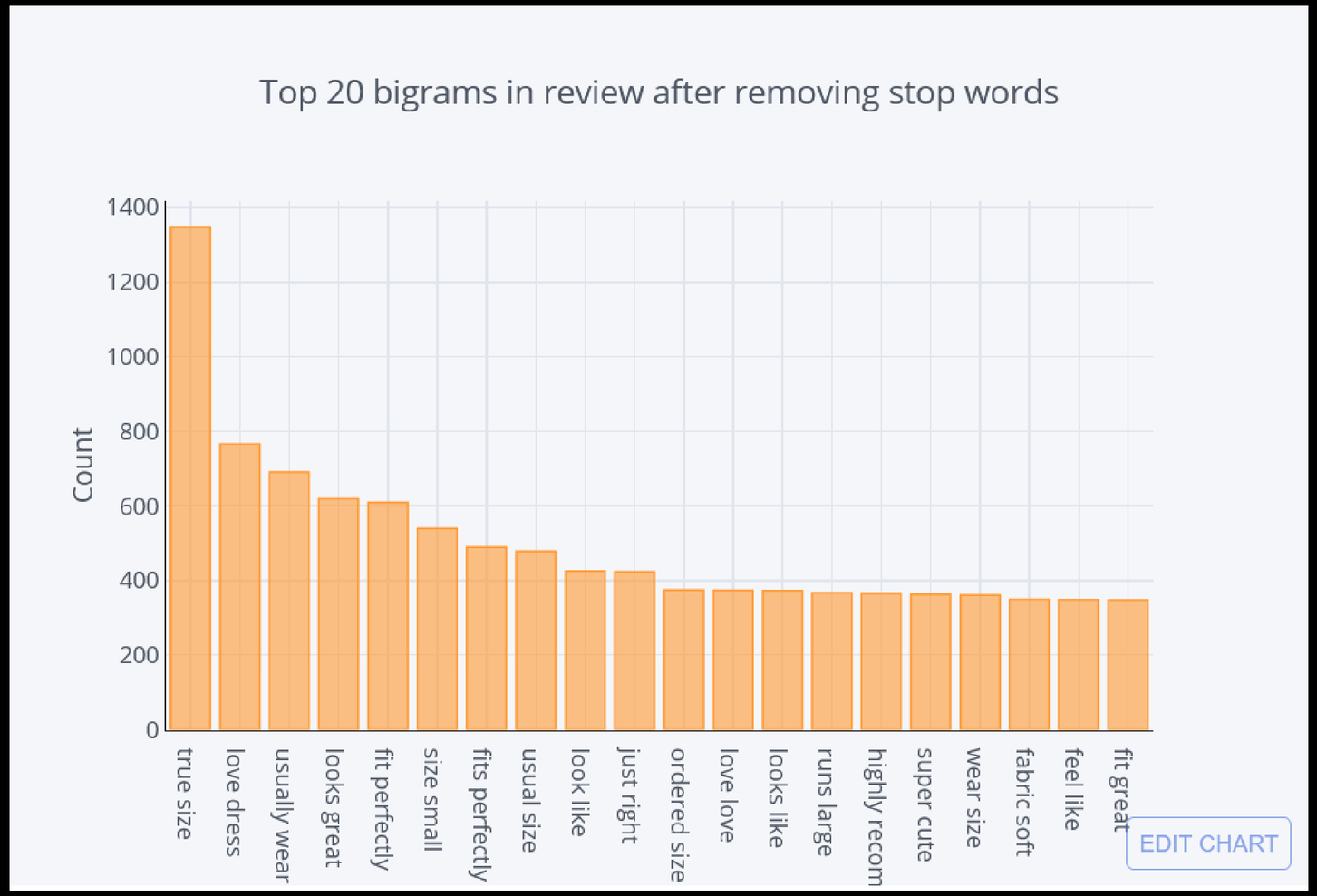
UNIGRAMS AFTER REMOVING STOPWORDS



BIGRAMS BEFORE REMOVING STOPWORDS

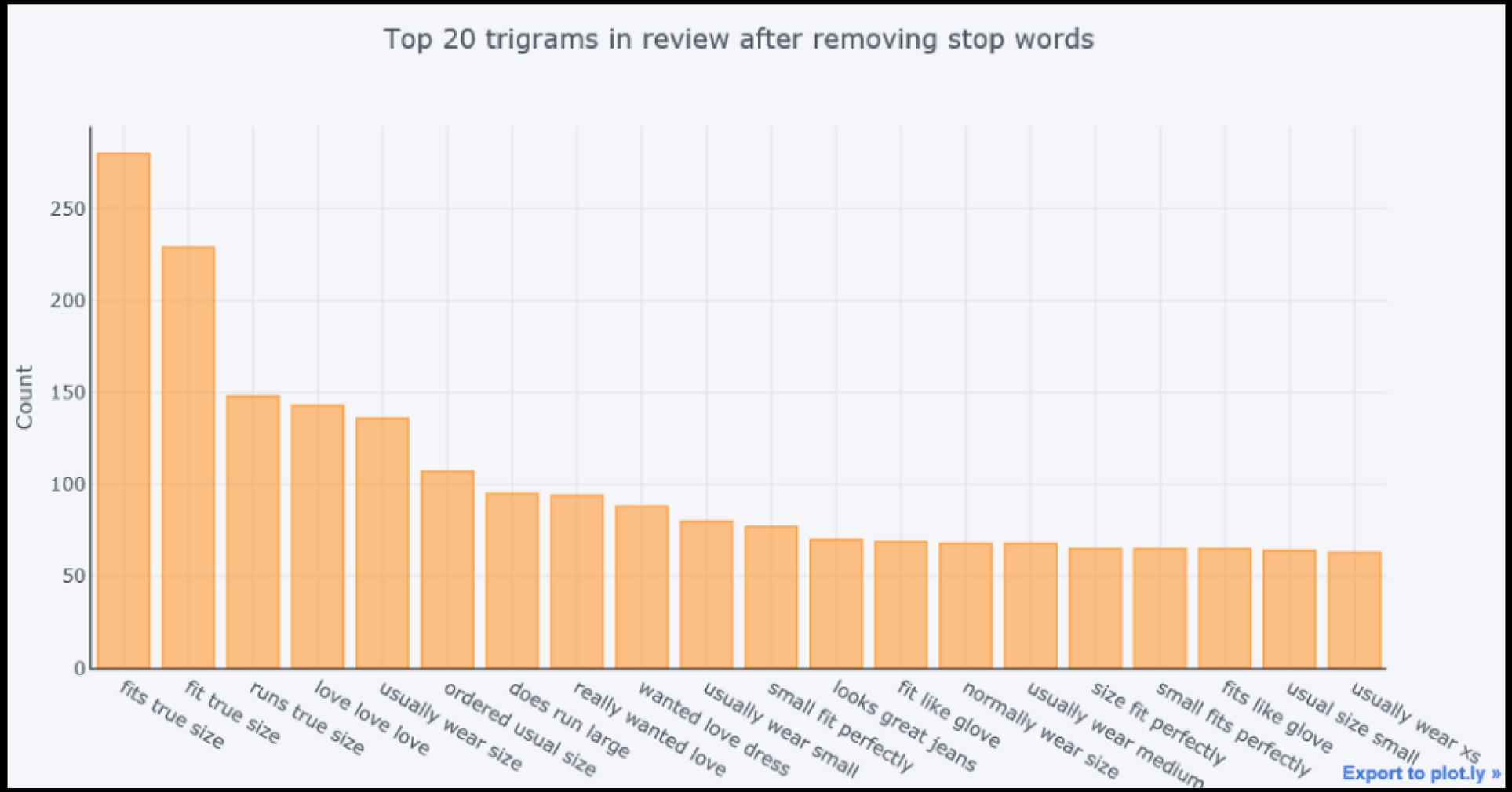


BIGRAMS AFTER REMOVING STOPWORDS

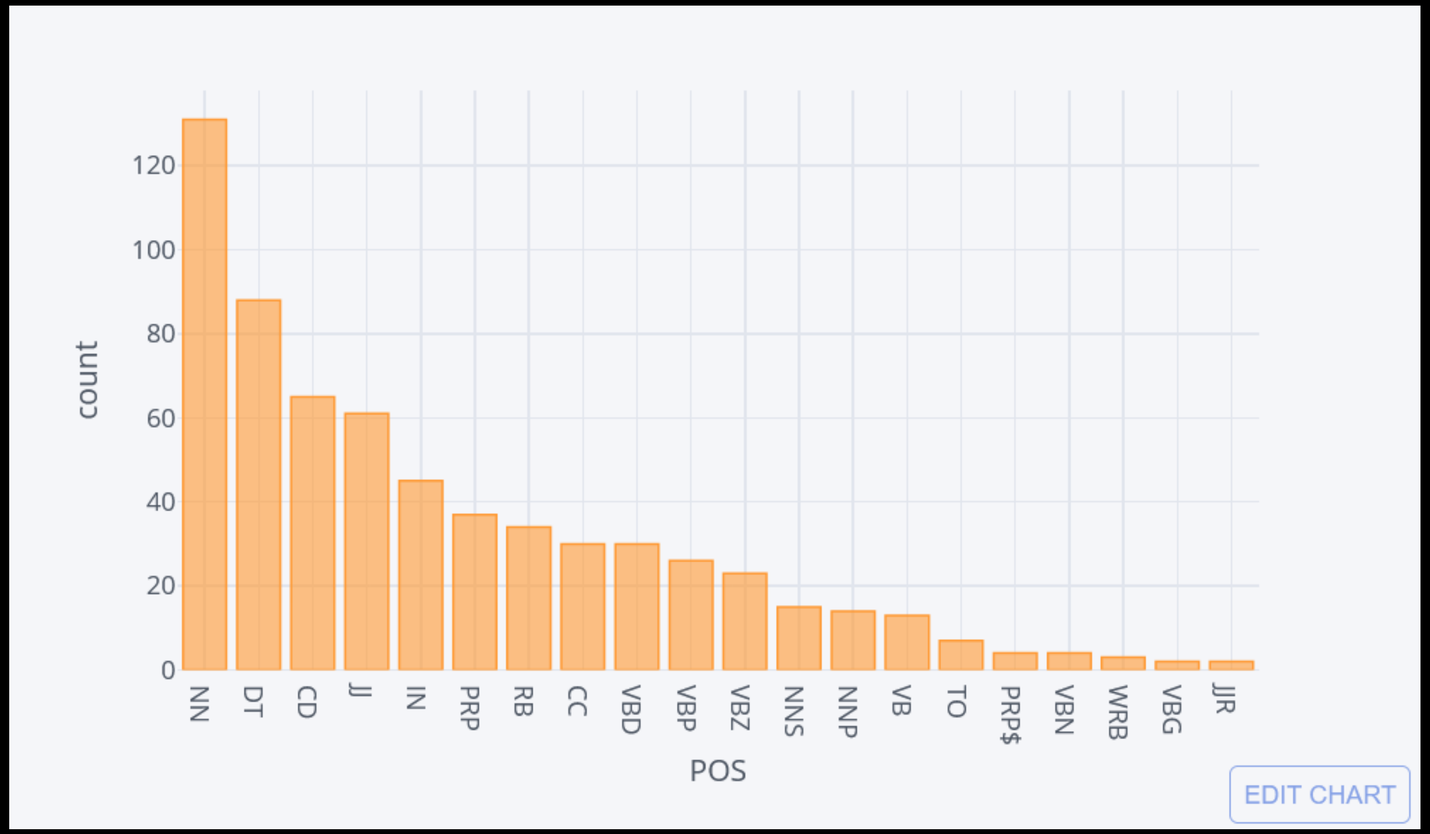


TRIGRAMS BEFORE REMOVING STOPWORDS

TRIGRAMS AFTER REMOVING STOPWORDS

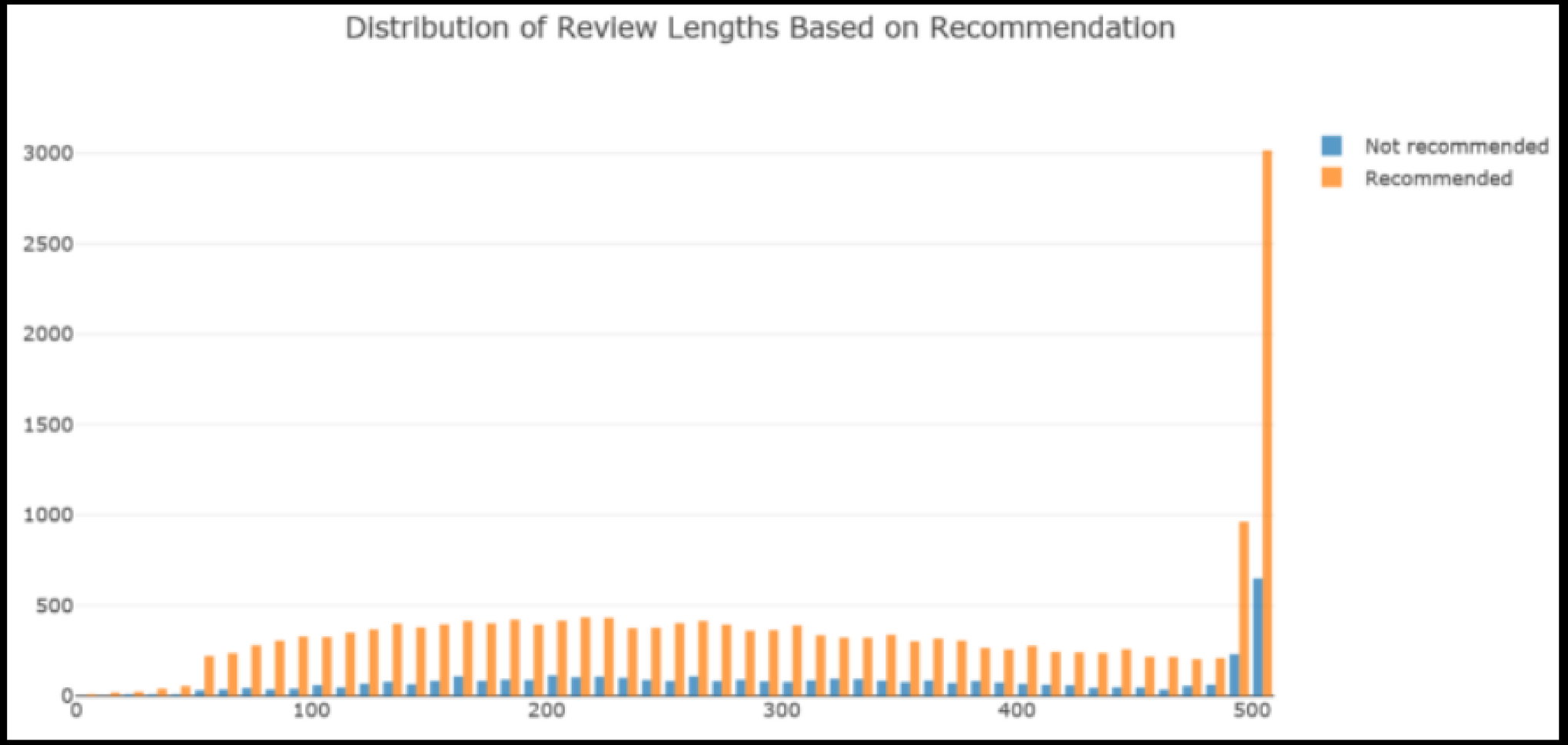


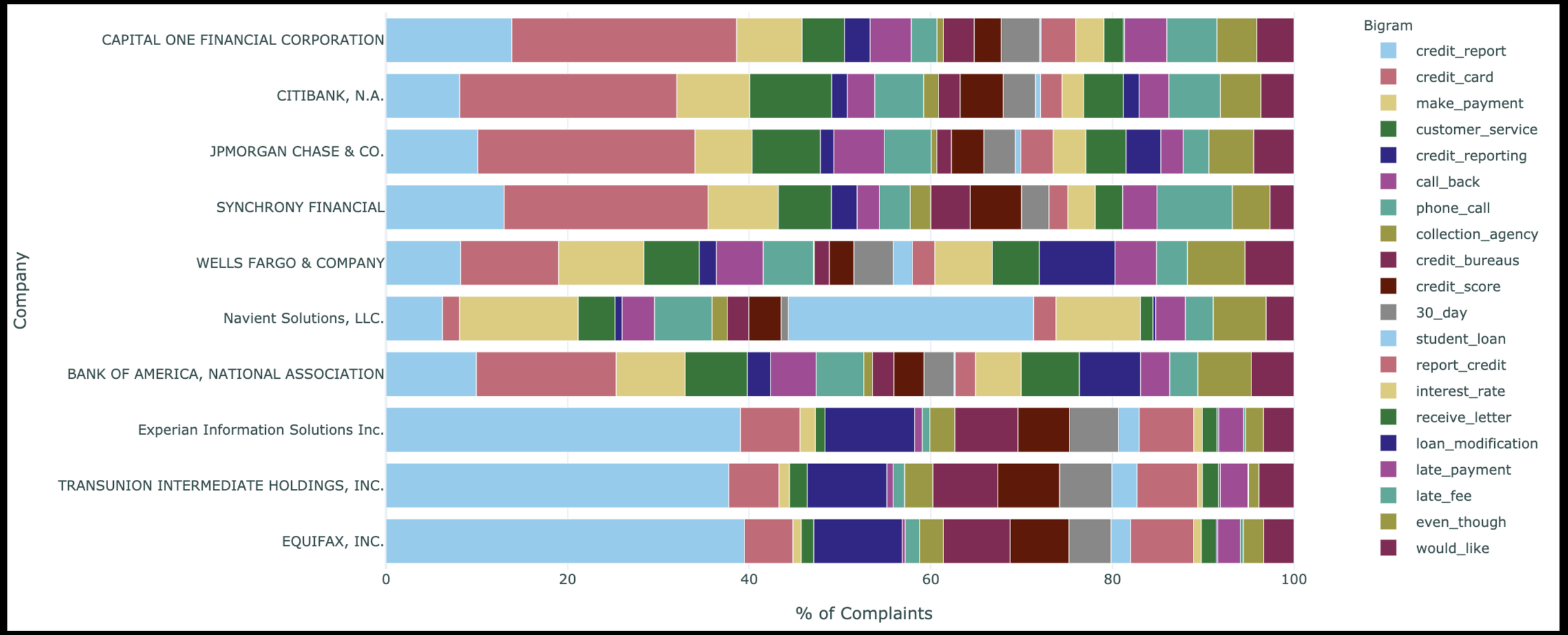
COUNTS OF PART-OF-SPEECH TAGS USING TEXTBLOB



See also:
https://textblob.readthedocs.io/en/dev/quickstart.html

COMBINATION OF POSSIBLY RELEVANT VARIABLES





Hwang, J. P. (2020, March 30). NLP visualisations for clear, immediate insights into text data and outputs. *Plotly*. https://medium.com/plotly/nlp-visualisations-for-clear-immediate-insights-into-text-data-and-outputs-9ebfab168d5b

IMPORTING DATA

- **From CSV**
  ```
  load_dataset("csv", data_files="my_file.csv")
  ```

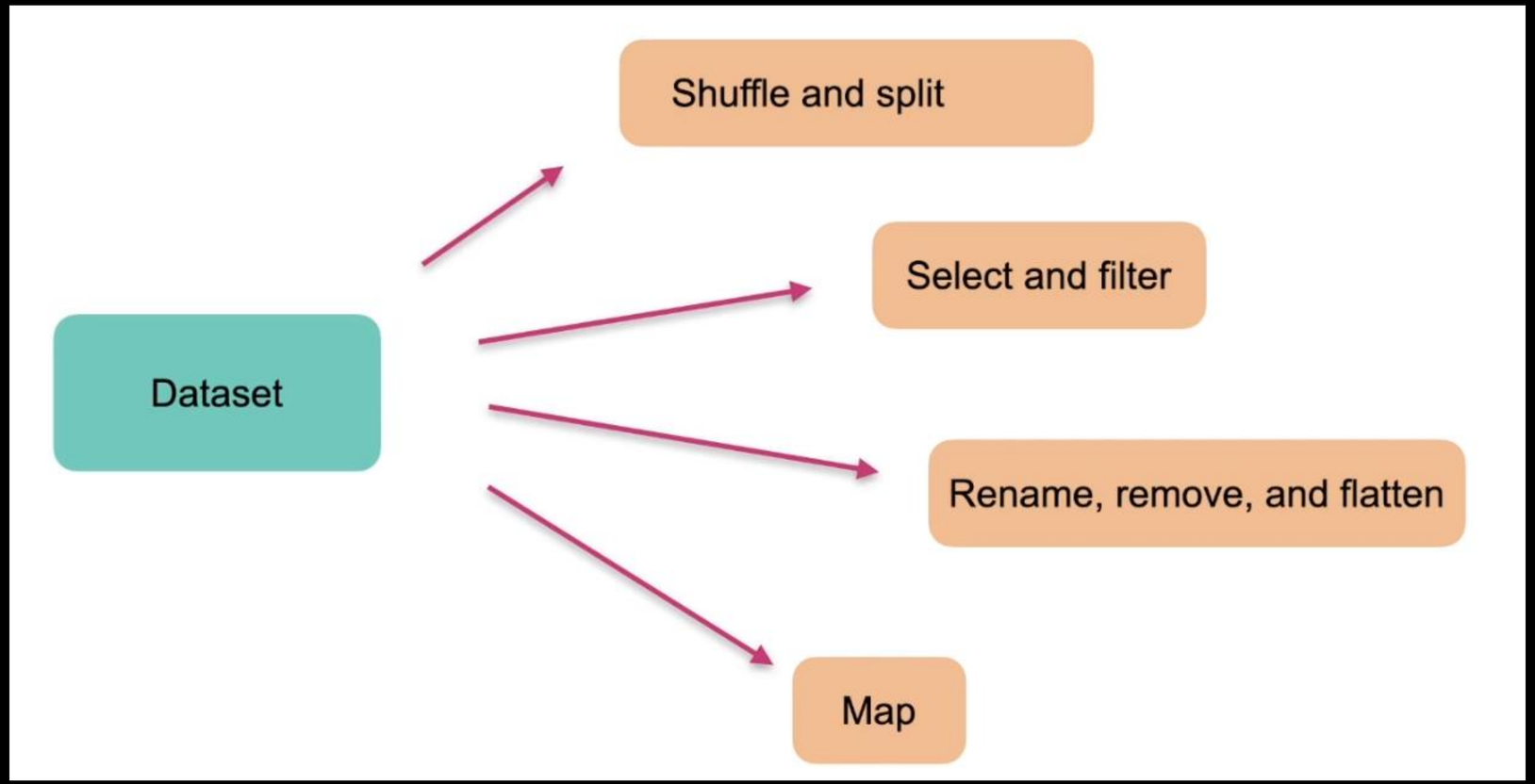
- **From JSON**
  ```
  load_dataset("json", data_files="my_file.jsonl")
  ```

- **From Pandas (Pickle)**
  ```
  load_dataset("pandas", data_files="my_dataframe.pkl")
  Dataset.from_pandas(my_dataframe)
  ```

APACHE ARROWS

- Backbone of the Huggingface Datasets library
- an efficient form of Pandas
- optimized for modern CPU and GPU hardware

DATASET METHODS



SAVING MODELS

GitHub, GitLab, Bitbucket, or a similar service using

- git and git LFS

Hugging Face Hub using

- huggingface_hub library (based on git and git FLS)
- push_to_hub API

HUGGING FACE HUB LIBRARY

```
# authentication
from huggingface_hub import notebook_login
notebook_login()

# saving via callback method
from transformers import PushToHubCallback
callback = PushToHubCallback(
    "bert-finetuned-mrpc",       save_strategy="epoch",
    tokenizer=tokenizer
)
model.fit(train_dataset, epochs=2, callbacks=callbacks)

# saving manually
model.push_to_hub("bert-finetuned-mrpc, commit="End of training")
```

PROJEKTPRÄSENTATION

- ca. 15 Minuten pro Projekt

- Der Inhalt sollte dem einer “Model Card” entsprechen:
  - Model description
  - Intended uses & limitations
  - How to use
  - Limitations and bias
  - Training data
  - Training procedure
  - Evaluation results

  Check this section here for more details.