In [1]:
```python
import pandas as pd

import re
from emoji import UNICODE_EMOJI
from textblob import TextBlob
import altair as alt
import numpy as np
from collections import Counter
import string

import nltk
nltk.download('vader_lexicon')
nltk.download('brown')
nltk.download('punkt')
nltk.download('stopwords')

from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords

import matplotlib.pyplot as plt
%matplotlib inline
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /home/jovyan/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
[nltk_data] Downloading package brown to /home/jovyan/nltk_data...
[nltk_data]   Package brown is already up-to-date!
[nltk_data] Downloading package punkt to /home/jovyan/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /home/jovyan/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

The data cleaning/manipulation technique/functions

In [2]:
```python
def extract_tags(text):
    return re.findall("#([a-zA-Z0-9_]{1,50})", text)

def extract_emoji(text):
    return [ch for ch in text if ch in UNICODE_EMOJI['en']]

def clean_tweet(txt):
    temp = re.sub("@[A-Za-z0-9_]+","", txt)
    temp1 = re.sub("#[A-Za-z0-9_]+","", temp)
    temp2 = re.sub(r"http\S+", "", temp1)

    result=''.join(i for i in temp2.lower() if (i.isalpha() or i==' '))
    return result

def word_list(tweet):

    lst = word_tokenize(tweet)
    lst1 = []
    stops = list(stopwords.words('english'))
    for w in lst:
        if w not in stops:
            lst1.append(w)

    return lst1

def sentiment(tweet):
    blob = TextBlob(tweet)

    return blob.sentiment.polarity


def get_date(date):

    return date[:10]

def get_hour(date):

    return date[11:13]
def get_10min(date):

    return date[14]+'0'

def get_min(date):

    return date[14:16]
```

```python
def firm_pos(score):
    if score >= 0.7:
        return 1
    else: return 0

def pos(score):
    if (score >= 0.25) & (score < 0.7):
        return 1
    else: return 0

def neutral(score):
    if (score >= -0.25) & (score < 0.25):
        return 1
    else: return 0

def neg(score):
    if (score > -0.7) & (score < -0.25):
        return 1
    else: return 0

def firm_neg(score):
    if score <= -0.7:
        return 1
    else: return 0
```

Import data, check duplicate and missing/incomplete data. Remove if exists

In [3]:
```python
df = pd.read_csv('Project Data/Simone Biles.csv')
df['id'].duplicated(keep='last').sum()
```

Out[3]:
```
0
```

In [4]:
```python
df.isnull().sum()
```

Out[4]:
```
id      0
date    2
text    2
dtype: int64
```

In [5]:
```python
df = df.dropna().reset_index()
df.drop(columns=['index'],inplace=True)
```

Apply the data cleaning/manipulation techniques on the data, we now have the used words, tags, emojis, sentiment score, and specific date/hour/min data.

In [6]:
```
df['tags']= df.apply(lambda row: extract_tags(row['text']), axis=1)
df['emojis']= df.apply(lambda row: extract_emoji(row['text']), axis=1)
df['clean_text']= df.apply(lambda row: clean_tweet(row['text']), axis=1)
df['words']= df.apply(lambda row: word_list(row['clean_text']), axis=1)
df['sentiment_score']= df.apply(lambda row: sentiment(row['clean_text']), axis=1)
df['day']= df.apply(lambda row: get_date(row['date']), axis=1)
df['hour']= df.apply(lambda row: get_hour(row['date']), axis=1)
df['10min']= df.apply(lambda row: get_10min(row['date']), axis=1)
df['min']= df.apply(lambda row: get_min(row['date']), axis=1)
df['POS']= df.apply(lambda row: firm_pos(row['sentiment_score']), axis=1)
df['pos']= df.apply(lambda row: pos(row['sentiment_score']), axis=1)
df['neu']= df.apply(lambda row: neutral(row['sentiment_score']), axis=1)
df['neg']= df.apply(lambda row: neg(row['sentiment_score']), axis=1)
df['NEG']= df.apply(lambda row: firm_neg(row['sentiment_score']), axis=1)

df.head()
```

Out[6]:

	id	date	text	tags	emojis	clean_text	words	sentiment_score	day	hour	10min	min	POS	pos
0	1418360000888647681	2021-07-22 23:59:10+00:00	the olympics r literally like tmrw??? anyways ...	[]	[]	the olympics r literally like tmrw anyways can...	[olympics, r, literally, like, tmrw, anyways, ...	0.000000	2021-07-22	23	50	59	0	0
1	1418359846253047810	2021-07-22 23:58:33+00:00	omg it’s the goat #SimoneBiles LOOK AT THE GOAT	[SimoneBiles]	[]	omg its the goat look at the goat	[omg, goat, look, goat]	0.000000	2021-07-22	23	50	58	0	0
2	1418359748500418573	2021-07-22 23:58:10+00:00	Simone Biles is a gymnast and an entertainer.I...	[]	[]	simone biles is a gymnast and an entertainerim...	[simone, biles, gymnast, entertainerim, glad, ...	0.166667	2021-07-22	23	50	58	0	0
3	1418359695681728515	2021-07-22 23:57:57+00:00	Just wanted to see the emoji. #SimoneBiles	[SimoneBiles]	[]	just wanted to see the emoji	[wanted, see, emoji]	0.000000	2021-07-22	23	50	57	0	0
4	1418359661028331526	2021-07-22 23:57:49+00:00	God, I’m not worthy enough to share a planet w...	[]	[]	god im not worthy enough to share a planet wit...	[god, im, worthy, enough, share, planet, simon...	-0.083333	2021-07-22	23	50	57	0	0

See the overall flow of tweet & sentiment

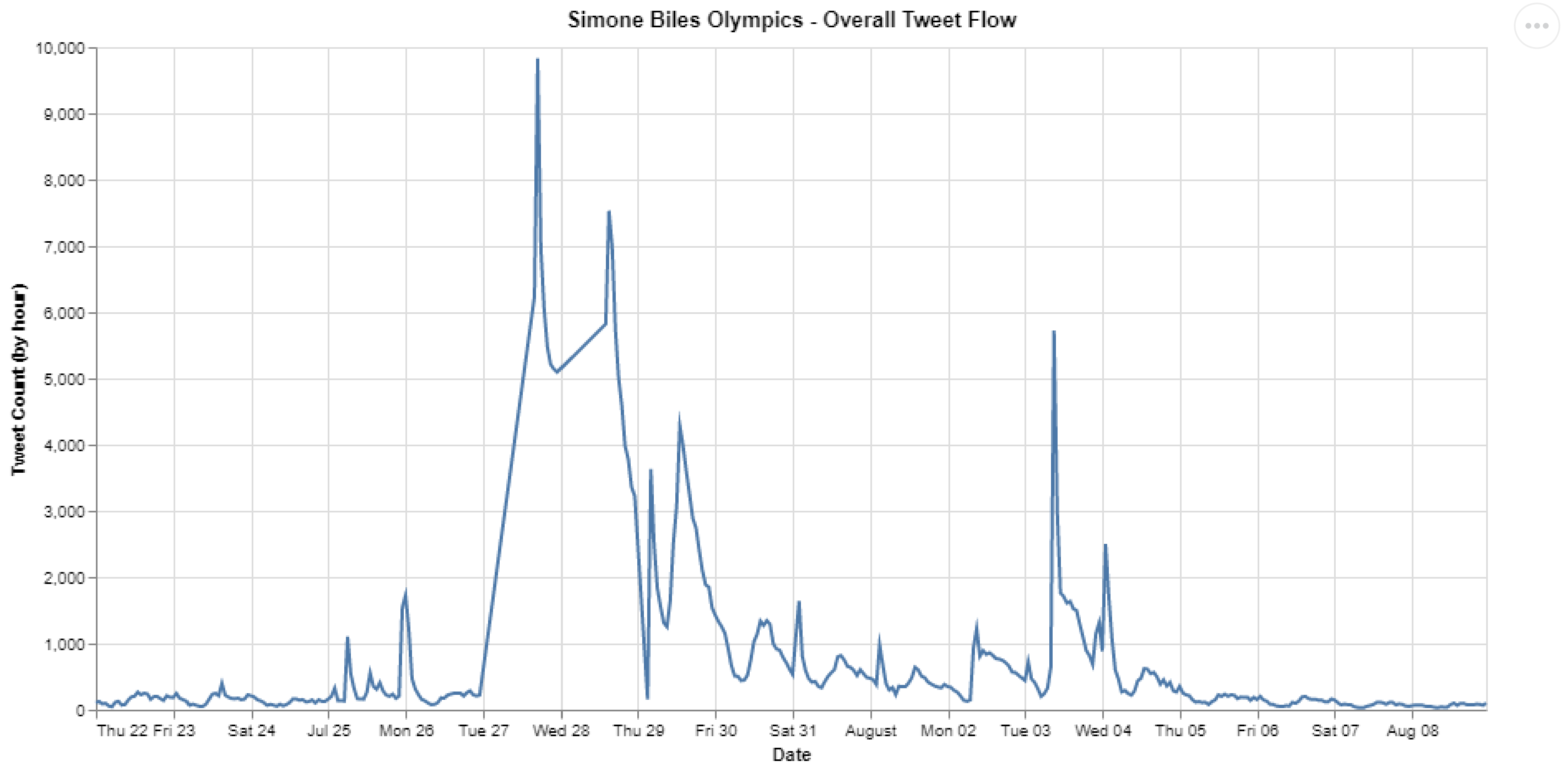
In [7]:
```
score = df.groupby(['day','hour']).agg([np.sum,np.size]).sentiment_score
score = score.reset_index()

score['date'] = score['day'] + ' ' + score['hour'] + ':00'
score[['6hr_sum','6hr_count']] = score.rolling(window=6,min_periods=1).sum()[['sum','size']]
score['6hr_avg'] = score['6hr_sum']/score['6hr_count']

score.head()
```

Out[7]:

	day	hour	sum	size	date	6hr_sum	6hr_count	6hr_avg
0	2021-07-22	00	28.321041	106.0	2021-07-22 00:00	28.321041	106.0	0.267180
1	2021-07-22	01	27.016179	125.0	2021-07-22 01:00	55.337220	231.0	0.239555
2	2021-07-22	02	18.577410	88.0	2021-07-22 02:00	73.914630	319.0	0.231707
3	2021-07-22	03	16.880703	97.0	2021-07-22 03:00	90.795333	416.0	0.218258
4	2021-07-22	04	12.141448	57.0	2021-07-22 04:00	102.936781	473.0	0.217625

In [8]:
```
alt.Chart(score).mark_line().encode(
    x=alt.X('date:T',title='Date'),
    y=alt.Y('size:Q',title='Tweet Count (by hour)')
).properties(width=840,height=400,title='Simone Biles Olympics - Overall Tweet Flow')
```

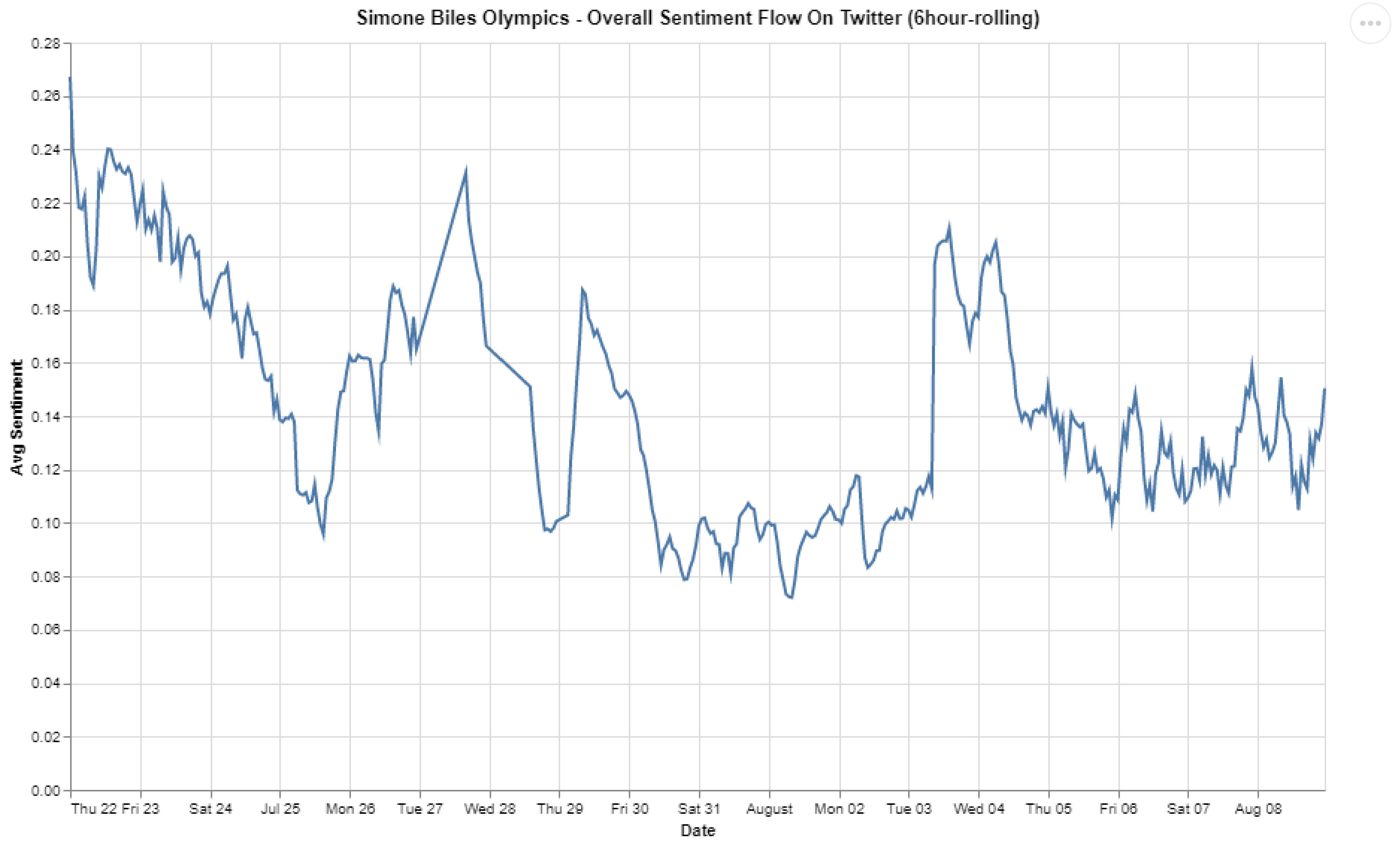
Out[8]:



Plot the sentiment flow

In [9]:
```
alt.Chart(score).mark_line().encode(
    x=alt.X('date:T',title='Date'),
    y=alt.Y('6hr_avg:Q',title='Avg Sentiment')
).properties(width=840,height=500,title='Simone Biles Olympics - Overall Sentiment Flow On Twitter (6hour-rolling)')
```

Out[9]:



In [10]:
```
flow = df.groupby(['day','hour']).mean()[['POS','pos','neu','neg','NEG']]
flow = flow.reset_index()
flow['date'] = flow['day'] + ' ' + flow['hour'] + ':00'
flow[['POSITIVE', 'positive', 'neutral', 'negative', 'NEGATIVE']] = flow.rolling(window=6,min_periods=1).mean()[['POS','pos','neu'

flow1 = pd.DataFrame()

dates = []
values = []
labels = []

for i in ['POSITIVE', 'positive', 'neutral', 'negative', 'NEGATIVE']:
    lst = []
    lst1 = []
    lst2 = list(flow.date.values)

    for j in range(len(flow)):
        lst.append(i)
        lst1.append(flow[i][j])

    dates += lst2
    labels += lst
    values += lst1

flow1['date'] = pd.Series(dates)
flow1['sentiment_label'] = pd.Series(labels)
flow1['percentage'] = pd.Series(values)
flow1.head(10)
```

Out[10]:

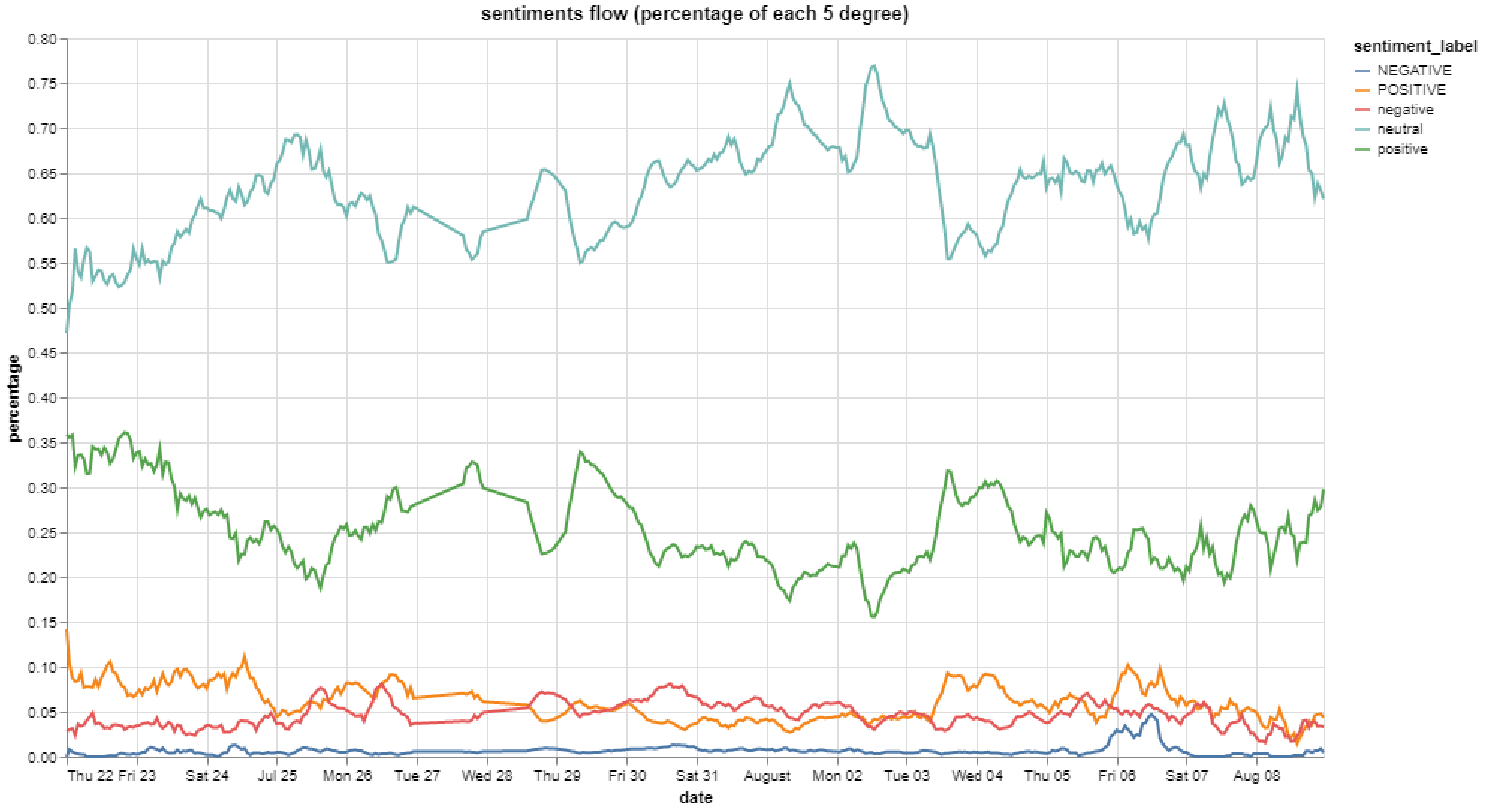
	date	sentiment_label	percentage
0	2021-07-22 00:00	POSITIVE	0.141509
1	2021-07-22 01:00	POSITIVE	0.102755
2	2021-07-22 02:00	POSITIVE	0.087443
3	2021-07-22 03:00	POSITIVE	0.083623
4	2021-07-22 04:00	POSITIVE	0.084442
5	2021-07-22 05:00	POSITIVE	0.093096
6	2021-07-22 06:00	POSITIVE	0.076951
7	2021-07-22 07:00	POSITIVE	0.078096

	date	sentiment_label	percentage
8	2021-07-22 08:00	POSITIVE	0.078016
9	2021-07-22 09:00	POSITIVE	0.076953

In [11]:

```
alt.Chart(flow1).mark_line().encode(
    x='date:T',
    y='percentage:Q',
    color='sentiment_label:N'
).properties(width=840,height=480,title='sentiments flow (percentage of each 5 degree)')
```

Out[11]:



Emoji/Tags

In [12]:
```python
# this return the top 50 most common items in the columns (emoji/tag/word)

def top_item(data,label):

    lst = []
    for i in data[label]:
        lst += i

    C = Counter(lst)
    top50 = C.most_common(50)
    count_df = pd.DataFrame(top50,columns = [label,'count'])

    return count_df
```

In [13]:

```python
c= top_item(df,'tags')
c

# the top 50 most frequently used tags      within the 'simone biles' tweets during the 2021 olympics
```

Out[13]:

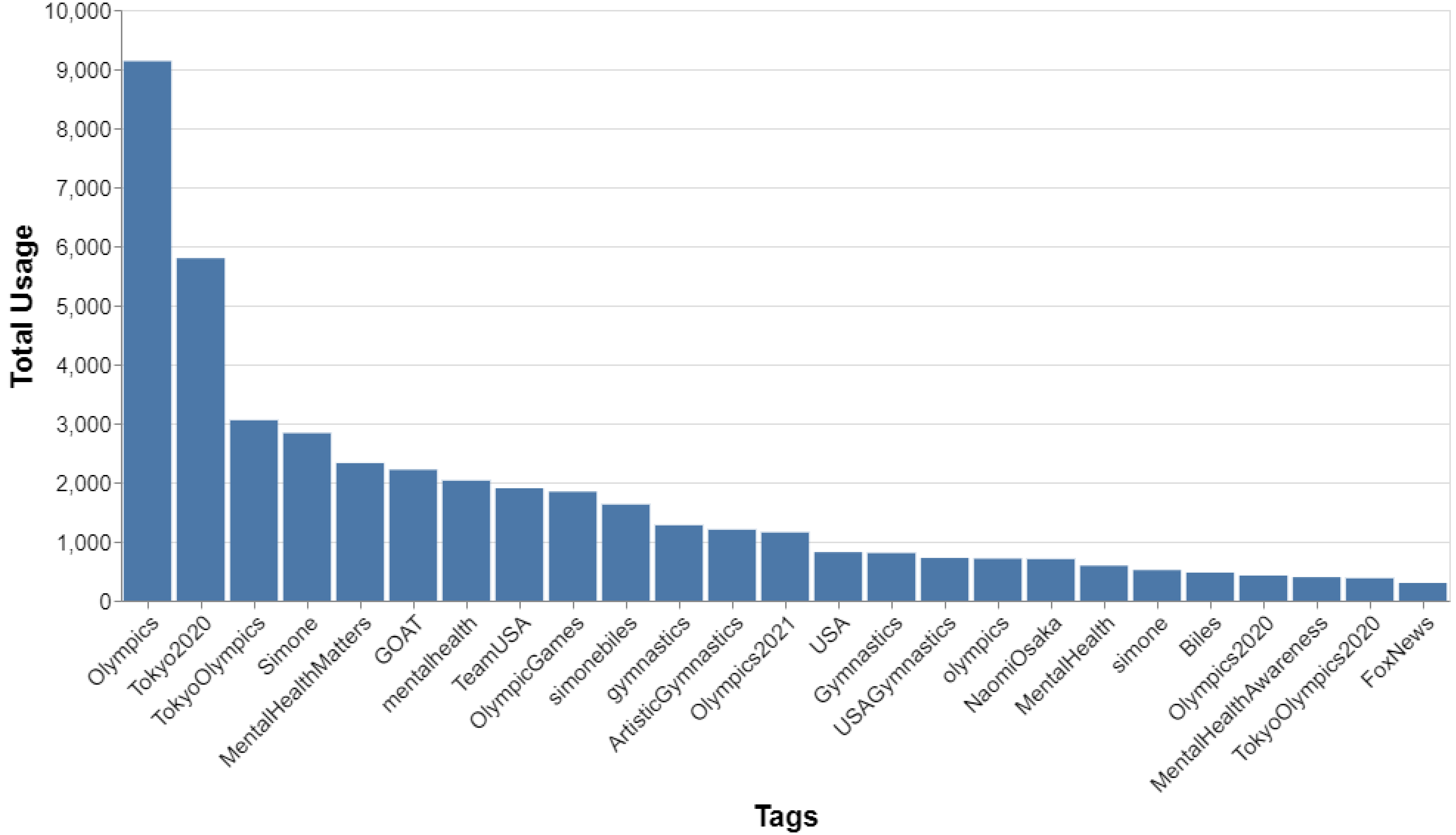
	tags	count
0	SimoneBiles	26354
1	Olympics	9137
2	Tokyo2020	5795
3	TokyoOlympics	3054
4	Simone	2835
5	MentalHealthMatters	2327
6	GOAT	2214
7	mentalhealth	2030
8	TeamUSA	1900
9	OlympicGames	1841
10	simonebiles	1631
11	gymnastics	1279
12	ArtisticGymnastics	1204
13	Olympics2021	1157
14	USA	822
15	Gymnastics	808
16	USAGymnastics	724
17	olympics	712
18	NaomiOsaka	705
19	MentalHealth	592
20	simone	518
21	Biles	476
22	Olympics2020	426
23	MentalHealthAwareness	398
24	TokyoOlympics2020	380

	tags	count
25	FoxNews	300
26	1	295
27	tokyo2020	284
28	Olympic	284
29	Mentalhealth	281
30	goat	269
31	usa	248
32	selfcare	245
33	PiersMorgan	242
34	SuniLee	236
35	SmartNews	233
36	Tokyo	224
37	SimonBiles	220
38	SimoneStrong	213
39	BlackGirlMagic	206
40	teamusa	199
41	NewsBreak	190
42	news	184
43	simonebilesgoat	182
44	SIMONEBILES	179
45	BREAKING	177
46	mentalhealthmatters	177
47	TokyoOlympics2021	171
48	sports	171
49	Tokyo2021	168

```
In [14]: c1=c[1:26]

         alt.Chart(c1).mark_bar().encode(
             x=alt.X('tags',sort=['count'],title='Tags',axis=alt.Axis(labelAngle=-45)),
             y=alt.Y('count',title='Total Usage')
         ).properties(width=900,height=400,title={
               "text": ["Most popular tags - Simone Biles Olympics"],
               "subtitle":["The top 25 most popular tags used within the tweets about Simone Biles during Olympics"]
             }).configure_axis(
             labelFontSize=15,
             titleFontSize=20
         ).configure_title(
             anchor='start',
             fontSize = 24,
             subtitleFontSize = 15
         )

Out[14]:
```

Most popular tags - Simone Biles Olympics

The top 25 most popular tags used within the tweets about Simone Biles during Olympics



Emoji

In [15]:
```python
count = top_item(df, 'emojis')
count
```

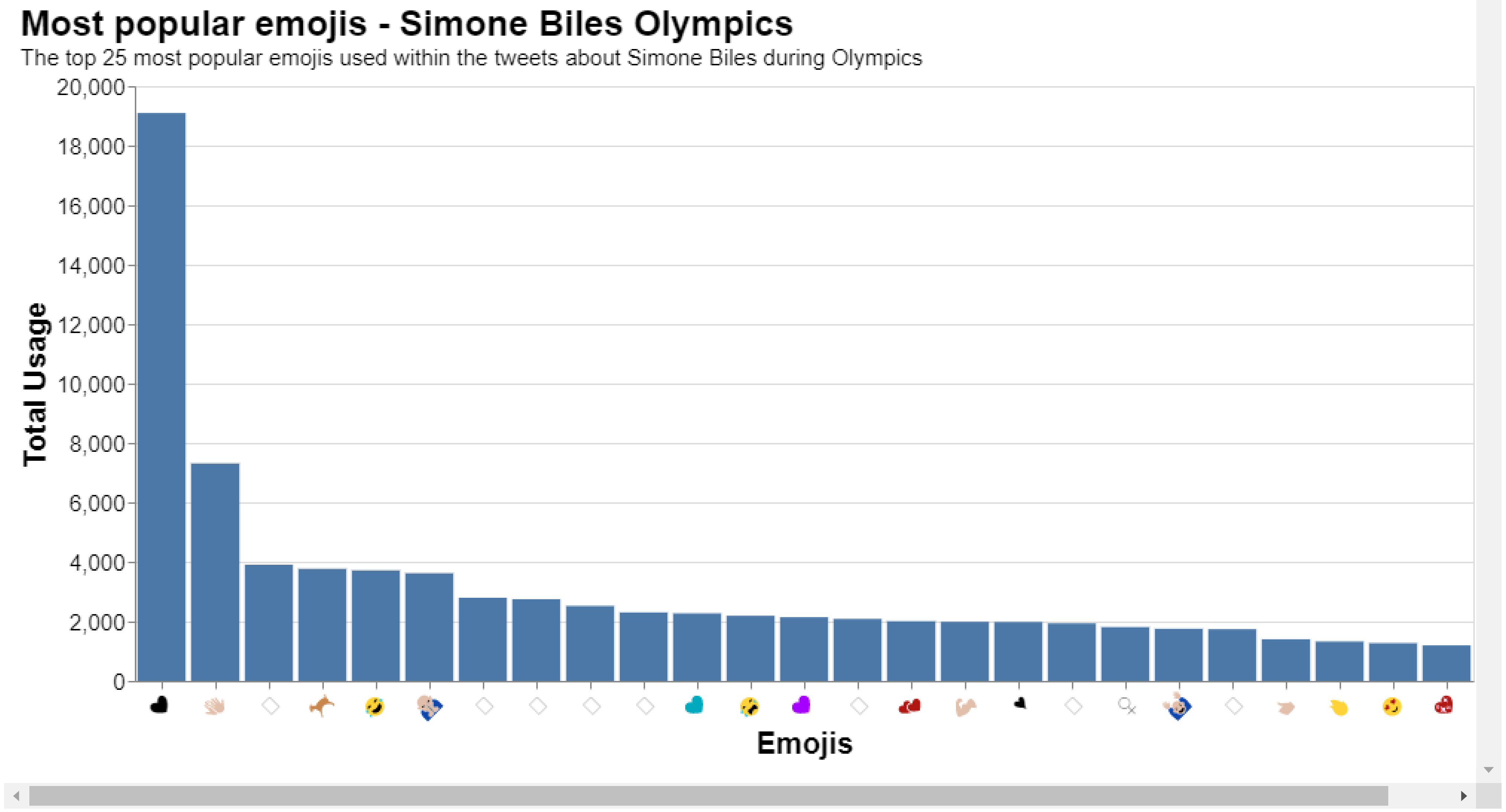
Out[15]:

	emojis	count
0	❤	19096
1	👏	7312
2		3903
3	🐐	3764
4	😂	3712
5	🙏	3616
6		2802
7		2753
8		2526
9		2307
10	💙	2275
11	😭	2195
12	💜	2146
13		2088
14	💕	2012
15	💪	1992
16	♥	1985
17		1939
18	♀	1814
19	🙌	1757
20		1743
21	👍	1404
22	🔥	1328
23	😍	1273
24	💖	1197

	emojis	count
25		1184
26	💯	1146
27		1122
28	♂	1106
29		1045
30		1003
31		991
32		975
33		886
34	✨	867
35		859
36	👑	838
37	💗	782
38		727
39	😚	711
40	✊	652
41	💜	647
42		638
43	👇	621
44		591
45	😊	549
46	🎉	549
47		509
48	💙	506
49		447

In [16]:
```
c1=count[:25]

alt.Chart(c1).mark_bar().encode(
    x=alt.X('emojis',sort=['count'],title='Emojis',axis=alt.Axis(labelAngle=-45)),
    y=alt.Y('count',title='Total Usage')
).properties(width=900,height=400,title={
      "text": ["Most popular emojis - Simone Biles Olympics"],
      "subtitle":["The top 25 most popular emojis used within the tweets about Simone Biles during Olympics"]
    }).configure_axis(
    labelFontSize=15,
    titleFontSize=20
).configure_title(
    anchor='start',
    fontSize = 24,
    subtitleFontSize = 15
)
```

Out[16]:



Specifically target the period after the withdrawl, see how Twitter react

In [17]:
```
df['Date'] = pd.to_datetime(df['date'])
mask = (df['Date'] > '2021-07-27 00:00') & (df['Date'] < '2021-07-28 23:59')
drawl = df.loc[mask].sort_values('Date')
drawl = drawl.reset_index()
drawl.drop(columns=['index','Date'],inplace=True)

drawl.head()
```

Out[17]:

	id	date	text	tags	emojis	clean_text	words	sentiment_score	day	hour	10min	min	POS	pos	neu
0	1420059487524765703	2021-07-27 16:32:19+00:00	@NYCMayor @Simone_Biles Omg what a hero! Remem...	[]	[]	omg what a hero remember the days we used to...	[omg, hero, remember, days, used, praise, real...	-0.100000	2021-07-27	16	30	32	0	0	1
1	1420059488002920459	2021-07-27 16:32:20+00:00	The conversation around Simone Biles is exactl...	[]	[]	the conversation around simone biles is exactl...	[conversation, around, simone, biles, exactly,...	0.525000	2021-07-27	16	30	32	0	1	0
2	1420059487931617281	2021-07-27 16:32:20+00:00	@Simone_Biles @lourdesgnavarro ❤ you, Simone ...	[]	[❤, ❤]	you simone and so proud of you	[simone, proud]	0.800000	2021-07-27	16	30	32	1	0	0
3	1420059491706433536	2021-07-27 16:32:20+00:00	@Simone_Biles @tonyposnanski You rock so hard!	[]	[]	you rock so hard	[rock, hard]	-0.291667	2021-07-27	16	30	32	0	0	0
4	1420059491484110848	2021-07-27 16:32:20+00:00	Logging in to say: good for Simone Biles.\nShe...	[]	[]	logging in to say good for simone bilesshe has...	[logging, say, good, simone, bilesshe, nothing...	0.075000	2021-07-27	16	30	32	0	0	1

```
In [18]: c = top_item(drawl,'tags')
         c
```

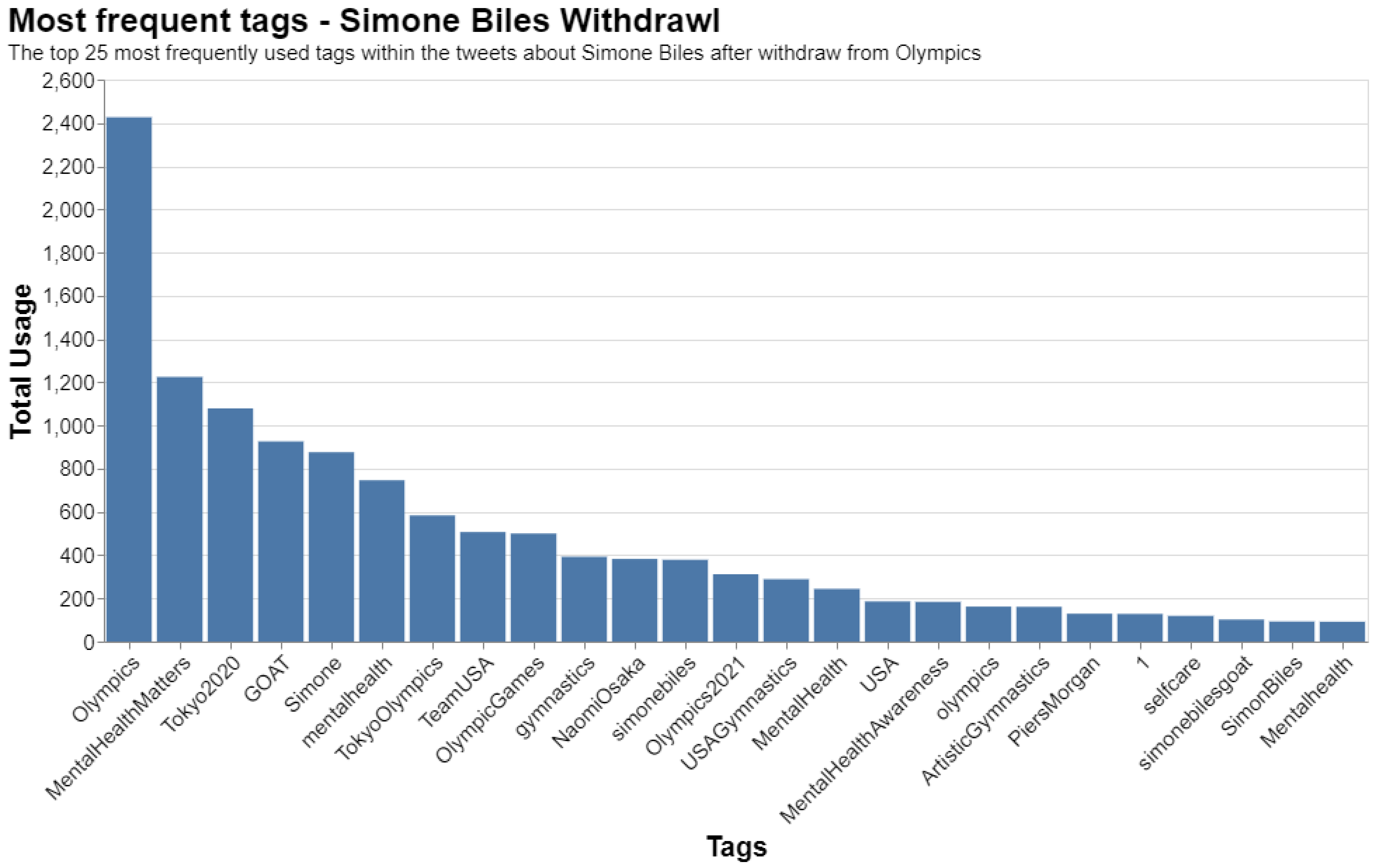
Out[18]:

	tags	count
0	SimoneBiles	9193
1	Olympics	2427
2	MentalHealthMatters	1225
3	Tokyo2020	1079
4	GOAT	926
5	Simone	876
6	mentalhealth	746
7	TokyoOlympics	583
8	TeamUSA	506
9	OlympicGames	499
10	gymnastics	393
11	NaomiOsaka	383
12	simonebiles	379
13	Olympics2021	312
14	USAGymnastics	289
15	MentalHealth	244
16	USA	186
17	MentalHealthAwareness	184
18	olympics	162
19	ArtisticGymnastics	161
20	PiersMorgan	129
21	1	128
22	selfcare	119
23	simonebilesgoat	102
24	SimonBiles	93
25	Mentalhealth	92

	tags	count
26	goat	87
27	Olympics2020	86
28	usa	85
29	simone	80
30	mentalhealthmatters	78
31	Biles	75
32	SimoneStrong	73
33	Tokyo	73
34	Gymnastics	67
35	Tokyo2021	67
36	tokyo2020	62
37	Olympic	61
38	Respect	57
39	MeghanMarkle	57
40	ActiBlizzWalkout	54
41	BlackGirlMagic	49
42	olympics2021	46
43	womensgymnastics	44
44	January6thCommission	44
45	TokyoOlympics2020	43
46	edit	42
47	fancam	42
48	followtrick	41
49	kpop	41

In [19]:
```
c1=c[1:26]

alt.Chart(c1).mark_bar().encode(
    x=alt.X('tags',sort=['count'],title='Tags',axis=alt.Axis(labelAngle=-45)),
    y=alt.Y('count',title='Total Usage')
).properties(width=900,height=400,title={
      "text": ["Most frequent tags - Simone Biles Withdrawl"],
      "subtitle":["The top 25 most frequently used tags within the tweets about Simone Biles after withdraw from Olympics"]
    }).configure_axis(
    labelFontSize=15,
    titleFontSize=20
).configure_title(
    anchor='start',
    fontSize = 24,
    subtitleFontSize = 15
)
```

Out[19]:


Most frequent tags - Simone Biles Withdrawl
The top 25 most frequently used tags within the tweets about Simone Biles after withdraw from Olympics
Total Usage
2,600
2,400
2,200
2,000
1,800
1,600
1,400
1,200
1,000
800
600
400
200
0
Olympics
MentalHealthMatters
Tokyo2020
GOAT
Simone
mentalhealth
TokyoOlympics
TeamUSA
OlympicGames
gymnastics
NaomiOsaka
simonebiles
Olympics2021
USAGymnastics
MentalHealth
USA
MentalHealthAwareness
olympics
ArtisticGymnastics
PiersMorgan
1
selfcare
simonebilesgoat
SimonBiles
Mentalhealth
Tags

```
In [ ]:
```