# Altair Exercises

This notebook will explore multiple different visualizations in Altair.

---

## Part 4

The following exercise is based on the article by FiveThirtyEight The Mayweather-McGregor Fight, As Told Through Emojis (https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/).

It leverages the dataset tweets (data/tweets.csv) Created by FiveThirtyEight with the Twitter Streaming API containing a sample of all the tweets that matched the search terms: #MayMac, #MayweatherMcGregor, #MayweatherVMcGregor, #MayweatherVsMcGregor, #McGregor and #Mayweather collected between 12:05 a.m. and 1:15 a.m. EDT, 12,118 that had emojis. Available on github (https://github.com/fivethirtyeight/data/tree/master/mayweather-mcgregor)

In [1]:
```
import pandas as pd
import numpy as np
import altair as alt
```

In [2]:
```
# enable correct rendering
alt.renderers.enable('default')

# uses intermediate json files to speed things up
alt.data_transformers.enable('json')
```

Out[2]:
```
DataTransformerRegistry.enable('json')
```

In [3]:
```
# load the tweets
tweets = pd.read_csv('../assets/tweets.csv')

# we're going to process the data in a couple of ways
# first, we want to know how many emojis are in each tweet so we'll c
# that counts them
tweets['emojis'] = tweets['text'].str.findall(r'[^\w\s.,"@\'?/#!$%\^&

# next, there are a few specific emojis that we care about, we're goi
# a column for each one and indicate how many times it showed up in t
boxer_emojis = ['☘️','🇮🇪','🍀','💸','🤑','💰','💵','😴','😂','🤣','
for emoji in boxer_emojis:
    # here's a different way to get the counts
    tweets[emoji] = tweets.text.str.count(emoji)

# For the irish pride vs the money team we want the numer
# of either ☘️, 🇮🇪 or 🍀 and 💸, 🤑, 💰 or 💵 for each
```

```
tweets['irish_pride'] = tweets['☘️'] + tweets['🇮🇪'] + tweets['☘']
tweets['money_team'] = tweets['💸'] + tweets['🤑'] + tweets['💰'] +
```

In [4]:
```
tweets.head()
```

Out[4]:

| | created_at | emojis | id | link | retw |
|---|---|---|---|---|---|
| **0** | 2017-08-27 00:05:34 | 1 | 901656910939770881 | https://twitter.com/statuses/901656910939770881 | |
| **1** | 2017-08-27 00:05:35 | 5 | 901656917281574912 | https://twitter.com/statuses/901656917281574912 | |
| **2** | 2017-08-27 00:05:35 | 2 | 901656917105369088 | https://twitter.com/statuses/901656917105369088 | |
| **3** | 2017-08-27 00:05:35 | 2 | 901656917747142657 | https://twitter.com/statuses/901656917747142657 | |
| **4** | 2017-08-27 00:05:35 | 2 | 901656916828594177 | https://twitter.com/statuses/901656916828594177 | |

5 rows × 25 columns

# The Mayweather-McGregor Fight, As Told Through Emojis

## We laughed, cried and cried some more.

*Original article available at FiveThirtyEight (https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/)*

By Dhrumil Mehta (https://fivethirtyeight.com/contributors/dhrumil-mehta/), Oliver Roeder (https://fivethirtyeight.com/contributors/oliver-roeder/) and Rachael Dottle (https://fivethirtyeight.com/contributors/rachael-dottle/)

Filed under Mayweather vs. McGregor (https://fivethirtyeight.com/tag/mayweather-vs-mcgregor/)

Get the data on GitHub (https://github.com/fivethirtyeight/data/tree/master/mayweather-mcgregor)

For the nearly 15,000 people in Las Vegas's T-Mobile Arena on Saturday night, and the millions more huddled around TVs across the world, the Floyd Mayweather–Conor McGregor fight was a roller coaster of emotions. They were anxious as pay-per-view technical problems (http://www.espn.com/boxing/story/_/id/20469815/floyd-mayweather-conor-mcgregor-delay-ppv-problems) pushed back the fight's start. They were full of anticipation when the combatants finally emerged after months of hype. They were surprised when McGregor held his own, or seemed to hold his own, for a couple of rounds. They were thrilled when Mayweather finally started fighting. And they were exhausted by the end.

How do we know all this? Emojis.

We were monitoring Twitter on fight night, pulling tweets that contained fight-related hashtags — those that included #MayweatherVsMcgregor, for example. In the end, we collected about 200,000 fight-related tweets, of which more than 12,000 contained emojis. (To be clear, that's a small enough sample that this emojinalysis might not make it through peer review.)[1]

In [5]:
```
# dictionary that will map emoji to percentage
percentages = {}

# find total emojies
total = tweets['emojis'].sum()

# for each emoji, figure out how prevalent it is
emojis = ['😂','🤣','🥊','👊','👏','🇮🇪','💪','🔥','😭','💰']
for emoji in emojis:
    percentages[emoji] = [round(tweets[emoji].sum() / total * 100,1)]

# create a data frame to hold this from the dictionary
percentages_df = pd.DataFrame.from_dict(percentages).T

# sort the dictionary
percentages_df = percentages_df.sort_values(by=[0], ascending = False)

# rename the columns
percentages_df = percentages_df.rename(columns={'index':'EMOJI', 0: 'P

# create a rank column based on position in the ordered list
percentages_df['rank'] = pd.Index(list(range(1,11)))

# modify the text
percentages_df['PERCENT_TEXT'] = percentages_df['PERCENT'].astype('str
```

In [6]:
```
percentages_df
```

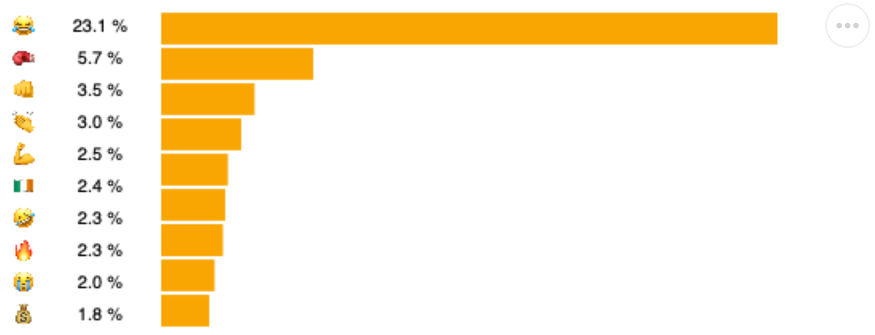
Out[6]:

| | EMOJI | PERCENT | rank | PERCENT_TEXT |
|---|---|---|---|---|
| **0** | 😂 | 23.1 | 1 | 23.1 % |
| **1** | 🥊 | 5.7 | 2 | 5.7 % |
| **2** | 👊 | 3.5 | 3 | 3.5 % |
| **3** | 👏 | 3.0 | 4 | 3.0 % |
| **4** | 💪 | 2.5 | 5 | 2.5 % |
| **5** | 🇮🇪 | 2.4 | 6 | 2.4 % |
| **6** | 🤣 | 2.3 | 7 | 2.3 % |
| **7** | 🔥 | 2.3 | 8 | 2.3 % |
| **8** | 😭 | 2.0 | 9 | 2.0 % |
| **9** | 💰 | 1.8 | 10 | 1.8 % |

In [7]:
```
# use percentages_df to recreate the visualization above
```

```
sort_pct = list(percentages_df.sort_values(by=['PERCENT'],ascending=Fa
sort_emoji = list(percentages_df.sort_values(by=['PERCENT'],ascending=

bars = alt.Chart(percentages_df).mark_bar().encode(
    x=alt.X(
        'PERCENT:Q',
        axis=None),
    y=alt.Y(
        'PERCENT_TEXT:N',
        axis=None,
        sort=sort_pct)
).properties(
    height=198
)

ranked_emoji = alt.Chart(percentages_df).mark_text(
).encode(
    y=alt.Y(
        'row_number:O',
        axis=None),
).transform_window(
    row_number='row_number()'
).transform_window(
    rank='rank(row_number)'
)

ranked_emoji

emojis = ranked_emoji.encode(text='EMOJI:N')
emoji_pcts = ranked_emoji.encode(text='PERCENT_TEXT:O')
tables = alt.hconcat(emojis, emoji_pcts)

emoji_dist = (tables | bars).configure_mark(
    color='#F9A602'
).configure_view(
    strokeWidth=0
)

emoji_dist
```

Out[7]:



There were the likely frontrunners for most-used emoji: the 🥊, the 👊, the 💪. But the emoji of the fight was far and away the 😂. ("Face with tears of joy.")[2]

> 1.2. That's certainly appropriate for this spectacle, but it should be noted that 😂 is also the most tweeted (http://emojitracker.com/) emoji generally.

Here's how the night unfolded, emoji-wise. (All of the charts below show them on a four-minute rolling average.)

For one thing, the fight was a sharply partisan affair. The majority of people in the arena appeared to be McGregor fans — he hails from Dublin and an Irish flag, worn cape-style, almost seemed like the evening's dress code. But other fans were members of TMT — The Money Team — and loyal to "Money" Mayweather. Twitter's loyalties came and went as the match progressed, with enthusiasm from either camp seemingly matching each fighter's success.

In [8]:
```
# Pre-Processing

# We're going to want to work with time objects so we need to make a c
# column (basically transforming the text in "created at"). It duplica
# the data but it will make things easier
tweets['datetime'] = pd.to_datetime(tweets['created_at'])
tweets = tweets.set_index('datetime')

# next we're going to creat a rolling average
# first for the money team
mdf = (tweets['money_team'].rolling('4Min').mean().groupby(pd.Grouper(
mdf['team'] = '💸🤑💰💵'
mdf = mdf.rename(columns={'money_team':'tweet_count'})

# next for the irish team
idf = (tweets['irish_pride'].rolling('4Min').mean().groupby(pd.Grouper
idf['team'] = '☘️☘️🇮🇪'
idf = idf.rename(columns={'irish_pride':'tweet_count'})

# now we'll combine our datasets
ndf = pd.concat([mdf,idf])
```

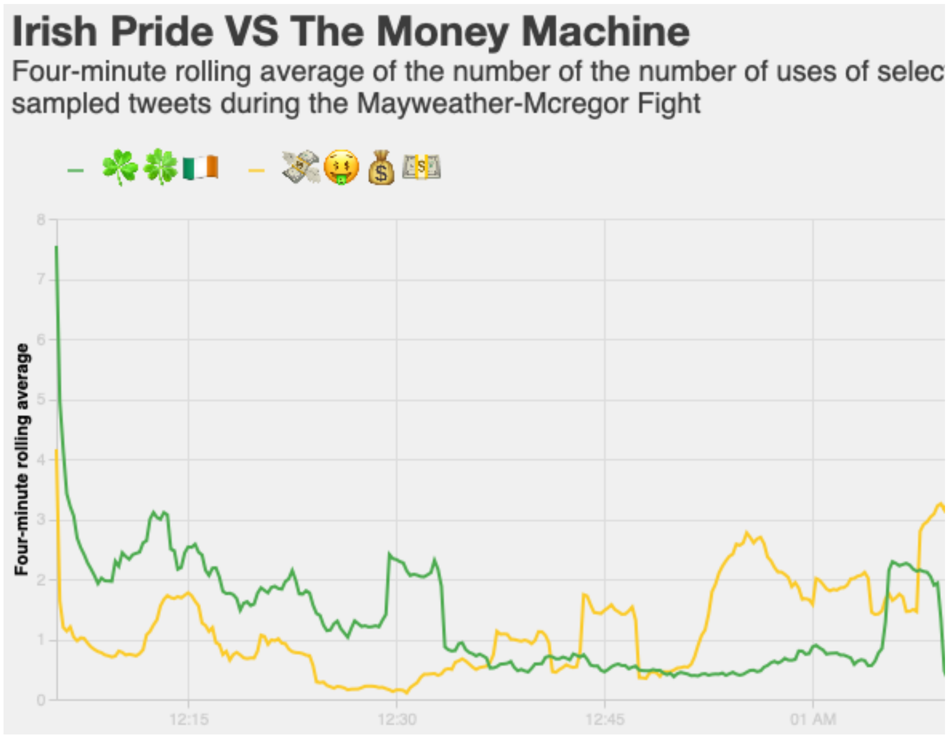
In [9]:
```
ndf.sample(5)
```

Out[9]:
```
                datetime  tweet_count      team
248  2017-08-27 01:07:30     1.459120  💸🤑💰💵
121  2017-08-27 00:35:45     0.720160    ☘️☘️🇮🇪
175  2017-08-27 00:49:15     0.436247    ☘️☘️🇮🇪
 56  2017-08-27 00:19:30     0.687007  💸🤑💰💵
  5  2017-08-27 00:06:45     1.035168  💸🤑💰💵
```

In [10]:
```
# we're also going to create an annotations
annotations = [['2017-08-27 00:10:00',4, 'Fight begins'],
               ['2017-08-27 00:22:00',5, 'McGregor does OK \nin the ea
               ['2017-08-27 00:53:00',4, "Mayweather takes \nover and
a_df = pd.DataFrame(annotations, columns=['date','count','note'])

# we're also going to create an annotation line
a_line = pd.DataFrame({
    'x': ['2017-08-27 00:15:00', '2017-08-27 00:15:00', '2017-08-27 00
    'y': [3.5, 2, 4.2, 3.9],
    'class': ['A', 'A', 'B', 'B']
})
a_line['x'] = pd.to_datetime(a_line['x'])
```

In [11]:
```
fight_lines = alt.Chart(ndf).mark_line().encode(
    x=alt.X('datetime:T',title=None,axis=alt.Axis(tickCount=5)), #bin=
    y=alt.Y('tweet_count:Q',title='Four-minute rolling average'),
    color=alt.Color('team', legend=alt.Legend(orient="top",title=None,
            domain=['☘️☘️🇮🇪', '💸🤑💰💵'],
            range=['#4BAB4F', '#FCCB28'])),
                range=['#1BAD1E', '#FCCB20'])),
)

annotations = alt.Chart(a_df).encode(
    x=alt.X('date:T',axis=None),
    y=alt.Y('count:Q',axis=None),
    text=alt.Text('note:O'),
).mark_text(lineBreak='\n',align='left',fontSize=16) #.properties(widt

fight_ano_line = alt.Chart(a_line).mark_line(color='black').encode(
    x='x',
    y='y',
    detail='class'
)


# (fight_lines + annotations + fight_ano_line ).configure(background='
fight_lines.configure(background='#F0F0F0').configure_view(
    # we don't want a stroke around the bars
    strokeWidth=0
).properties(
    title = alt.TitleParams(text = "Irish Pride VS The Money Machine",
                            subtitle = ["Four-minute rolling average o
                                        "sampled tweets during the May
                            font = 'Helvetica Neue',
                            fontSize = 26,
                            color = '#3C3C3C',
                            subtitleColor='#3C3C3C',
                            subtitleFontSize = 18,
                            anchor='start',
                            offset=20,
                            ),
    # set the dimensions of the visualization
    width=600,
    height=300
).configure_axis(labelColor='#cccccc',domainColor='#cccccc',tickColor=
```

Out[11]:



To the surprise of many (of the neutral and pro-Mayweather viewers, anyway) McGregor won the first round. The next couple were washes, and a quarter of the way into the [scheduled 12 rounds (https://www.nytimes.com/2017/08/26/sports/mayweather-mcgregor.html)](https://www.nytimes.com/2017/08/26/sports/mayweather-mcgregor.html) … the Irish underdog may have been winning! The Irish flags and shamrocks followed on Twitter. Things slowly (perhaps even 😴ly) turned around as one of the best pound-for-pound boxers in history took control of the man making his pro debut — an outcome which was predicted by precisely everyone. Out came the emoji money bags.

By the sixth round, it seemed like only a matter of time until the old pro dismantled the newcomer. By the ninth it was clear Mayweather was going for the knockout. It came soon thereafter. Mayweather unleashed a vicious flurry of punches in the 10th and the ref stepped in, declaring Mayweather the victor and saving McGregor, who was somehow still on his feet, from further damage.

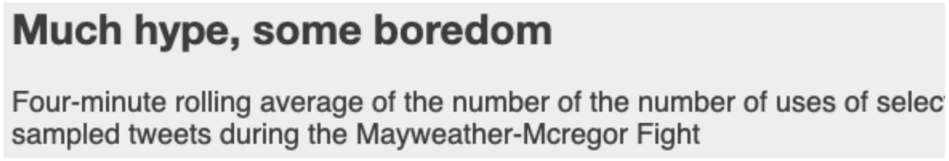
In [12]:
```
# We're going to want to work with time objects so we need to make a c
# column (basically transforming the text in "created at"). It duplica
# the data but it will make things easier
tweets['datetime'] = pd.to_datetime(tweets['created_at'])
tweets = tweets.set_index('datetime')

# next we're going to creat a rolling average
# first for fire
firedf = (tweets['🔥'].rolling('4Min').mean().groupby(pd.Grouper(freq
firedf['sentiment'] = '🔥'
firedf = firedf.rename(columns={'🔥':'tweet_count'})

# next for the snooze
snoozedf = (tweets['😴'].rolling('4Min').mean().groupby(pd.Grouper(fr
snoozedf['sentiment'] = '😴'
snoozedf = snoozedf.rename(columns={'😴':'tweet_count'})

# now we'll combine our datasets
hb = pd.concat([firedf,snoozedf])
hb.sample(5)
```

Out[12]:

| | datetime | tweet_count | sentiment |
|---|---|---|---|
| **53** | 2017-08-27 00:18:45 | 2.199223 | 🔥 |
| **207** | 2017-08-27 00:57:15 | 0.220710 | 😴 |
| **76** | 2017-08-27 00:24:30 | 0.281969 | 🔥 |
| **125** | 2017-08-27 00:36:45 | 0.682989 | 🔥 |
| **248** | 2017-08-27 01:07:30 | 0.187875 | 😴 |

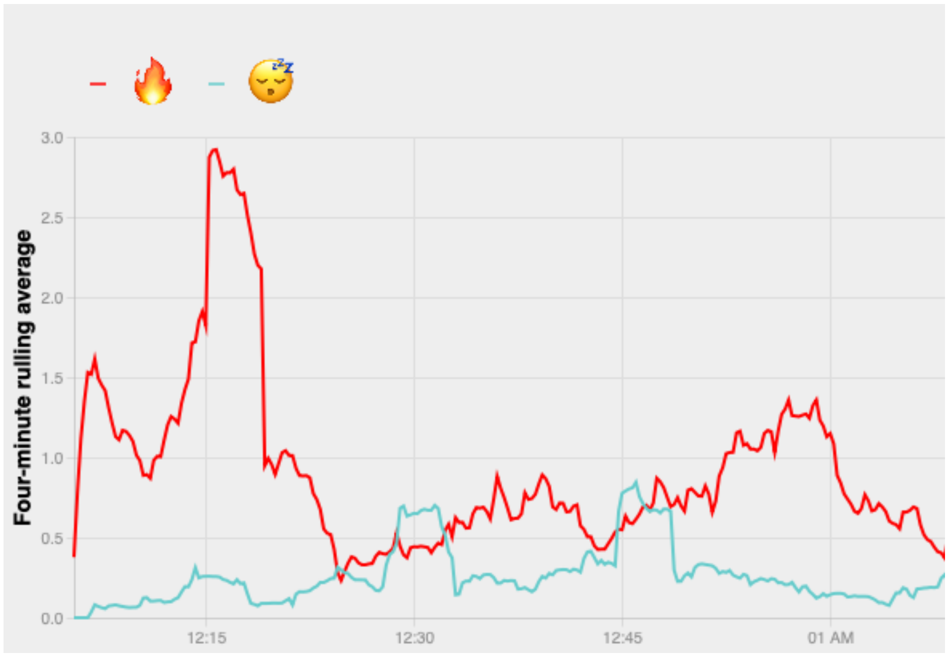
In [13]:
```
range_ = ['red', '#66CCCC']
annot = pd.DataFrame({'x' : ['2017-08-27 00:15:00', '2017-08-27 00:15:
                      'y' : [0.4, 0.9, 0.75, 1.25, 0.9, 1.25],
                      'class' : ['A', 'A', 'B', 'B', 'C', 'C']
                     })
annot['x'] = pd.to_datetime(annot['x'])

# we're also going to create an annotations data frame to help you
annotations = [['2017-08-27 00:10:00',1, 'Fight begins'],
               ['2017-08-27 00:33:00',3.5, 'Mayweather \ntakes control
sf_a_df = pd.DataFrame(annotations, columns=['date','count','note'])


lines = alt.Chart(hb).mark_line(
).encode(
    x=alt.X(
        'datetime',
        axis=alt.Axis(
            title='',
        tickMinStep=15,
        tickCount=4)
        tickCount=4)
    ),
    y=alt.Y(
        'tweet_count:Q',
        axis=alt.Axis(
            title='Four-minute rulling average',
            tickMinStep=0.5,
            tickCount=5)
    ),
    color=alt.Color(
        'sentiment:N',
        scale=alt.Scale(
            range=range_
        ))
)

# annotations = alt.Chart(annot).mark_line().encode(
#     x='x',
#     y='y',
#     detail='class'
# )

# hb_annotations = alt.Chart(sf_a_df).encode(
#     x=alt.X('date:T',axis=None),
#     y=alt.Y('count:Q',axis=None),
#     text=alt.Text('note:O'),
# ).mark_text(lineBreak='\n',align='left',fontSize=16)

final3 = lines

team = final3.configure(
    background='#eeeeee'
).configure_axis(
    labelFontSize=10,
    labelOpacity=0.4,
    tickOpacity=0.2,
    domainOpacity=0.2,
    titleFontSize=14
).configure_title(
    anchor='start',
    font='Helvetica',
    fontSize=24,
    fontWeight='bold',
    dy=-20
).configure_legend(
    labelFontSize=30,
    orient='top',
    title=None,
    titlePadding=10
).properties(
    title=alt.TitleParams(text = "Much hype, some boredom",
                          subtitle = ["Four-minute rolling average of
                                      "sampled tweets during the Maywe
                          font = 'Helvetica Neue',
                          fontSize = 26,
                          color = '#3C3C3C',
                          subtitleColor='#3C3C3C',
                          subtitleFontSize = 18,
                          anchor='start',
                          offset=20,
                         ),
    width=600,
    height=300
)

team
```

Out[13]:



**Much hype, some boredom**

Four-minute rolling average of the number of the number of uses of selec
sampled tweets during the Mayweather-Mcregor Fight



**Yet Another Visualization**

```
In [14]: # We're going to want to work with time objects so we need to make a c
         # column (basically transforming the text in "created at"). It duplica
         # the data but it will make things easier
         tweets['datetime'] = pd.to_datetime(tweets['created_at'])
         tweets = tweets.set_index('datetime')

         tears = tweets.copy()
         tears = tears.resample('1s').sum()
         tears  = tears[(tears['😭']>0) | (tears['🤣']>0)]

         # We're going to creat a rolling average
         cry = tears['😭'] .rolling('4Min').mean().reset_index()
         cry['emoji'] = '😭'
         cry = cry.rename(columns={'😭':'tweet_count'})

         laugh = tears['🤣'].rolling('4Min').mean().reset_index()
         laugh['emoji'] = '🤣'
         laugh = laugh.rename(columns={'🤣':'tweet_count'})

         # now we'll combine our datasets
         cl_df = pd.concat([cry,laugh])
         cl_df.head()
```

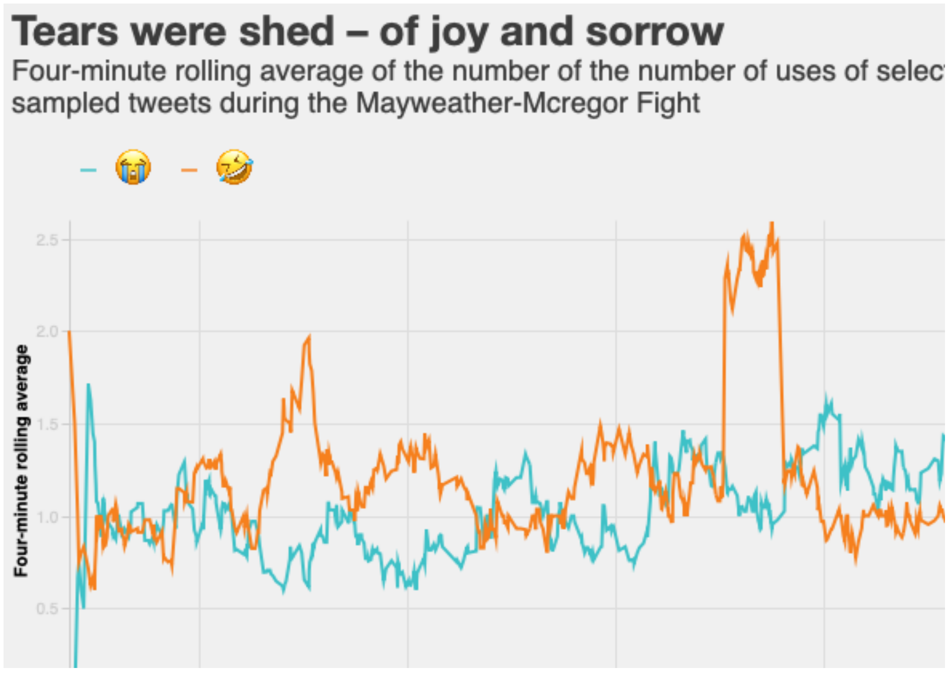
Out[14]:

| | datetime | tweet_count | emoji |
|---|---|---|---|
| **0** | 2017-08-27 00:05:35 | 0.000000 | 😭 |
| **1** | 2017-08-27 00:05:59 | 0.000000 | 😭 |
| **2** | 2017-08-27 00:06:12 | 0.666667 | 😭 |
| **3** | 2017-08-27 00:06:17 | 0.750000 | 😭 |
| **4** | 2017-08-27 00:06:27 | 0.600000 | 😭 |

```
In [15]: # we're also going to create an annotations data frame to help you
         annotations = [['2017-08-27 00:10:00',2.25, 'Fight begins'],
                        ['2017-08-27 00:30:00',2.25, 'McGregor \nimpresses \nea
                        ['2017-08-27 00:50:00',.25, 'Fight ends']]
         cl_a_df = pd.DataFrame(annotations, columns=['date','count','note'])

         # we're also going to create an annotation line
```

```
a3_line = pd.DataFrame({
    'x': ['2017-08-27 00:15:00', '2017-08-27 00:15:00', '2017-08-27 00
    'y': [2.1, 1.5, 2, 1.6,.4,.9],
    'class': ['A', 'A', 'B', 'B','C','C']
})
a3_line['x'] = pd.to_datetime(a3_line['x'])

cl_lines = alt.Chart(cl_df).mark_line().encode(
    x=alt.X('datetime:T',title=None,axis=alt.Axis(tickCount=5)), #bin=
    y=alt.Y('tweet_count:Q',title='Four-minute rolling average'),
    color=alt.Color('emoji', legend=alt.Legend(orient="top",title=None
            domain=['😭', '🤣'],
            range=['#3EC1C7', '#F6801D'])),
)

# cl_annotations = alt.Chart(cl_a_df).encode(
#     x=alt.X('date:T',axis=None),
#     y=alt.Y('count:Q',axis=None),
#     text=alt.Text('note:O'),
# ).mark_text(lineBreak='\n',align='left',fontSize=16) #.properties(wi

# ano3_line = alt.Chart(a3_line).mark_line(color='black').encode(
#     x='x',
#     y='y',
#     detail='class'
# )

# (cl_lines + cl_annotations + ano3_line ).configure(background='#F0F0
cl_lines.configure(background='#F0F0F0').configure_view(
    # we don't want a stroke around the bars
    strokeWidth=0
).properties(
    title = alt.TitleParams(text = "Tears were shed – of joy and sorro
                            subtitle = ["Four-minute rolling average o
                                        "sampled tweets during the May
                            font = 'Helvetica Neue',
                            fontSize = 26,
                            color = '#3C3C3C',
                            subtitleColor='#3C3C3C',
                            subtitleFontSize = 18,
                            anchor='start',
                            offset=20,
                            ),
    # set the dimensions of the visualization
    width=600,
    height=300
).configure_axis(labelColor='#cccccc',domainColor='#cccccc',tickColor=
```

Out[15]:



Exercise adapted and modified from UMSI homework assignment for SIADS 522.