Before you turn this problem in, make sure everything runs as expected. First, **restart the kernel** (in the menubar, select Kernel→Restart) and then **run all cells** (in the menubar, select Cell→Run All).

Make sure you fill in any place that says `YOUR CODE HERE` or "YOUR ANSWER HERE", as well as your name and collaborators below:

```
In [1]: NAME = "Liwen Huang"
```

Information Visualization I

School of Information, University of Michigan

Week 4:

- Data Types
- Design

Assignment Overview

This assignment's objectives include:

- Review, reflect on, and apply the concepts of encoding different data types. Given a visualization, justify different encodings for certain datatypes.
- Review, reflect on, and apply good design decisions in a visualization. Given a visualization critique design decisions according to principles like Tufte's data-ink ratio, graphical integrity, chart junk, Munzner's rules of thumb, etc.
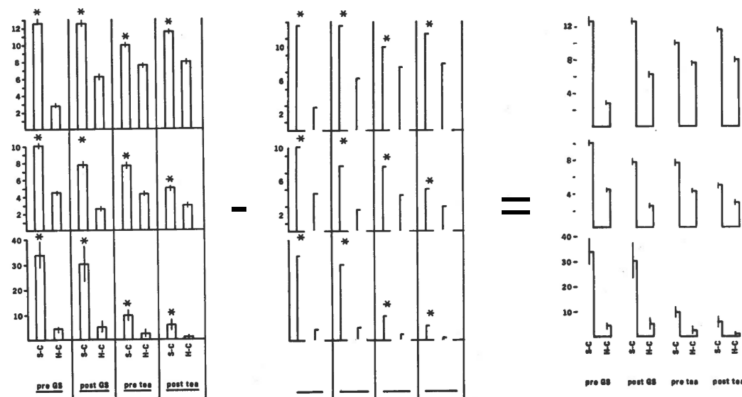
Same information, less ink

- Recreate, propose new and alternative visualizations using Altair (https://altair-viz.github.io/)

The total score of this assignment will be 100 points consisting of:

- Case study reflection: The Mayweather-McGregor Fight, As Told Through Emojis (30 points)
- Altair programming exercise (70 points)

Resources:

- Article by Five Thirty Eight (https://fivethirtyeight.com) available online (https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/) (Hickey, Koeze, Dottle, Wezerek 2017)
- Datasets from Five Thirty Eight, we have download a subset of these datasets to ./assets (assets) but the original can be found on Five Thirty Eight Mayweather vs McGregor (https://github.com/fivethirtyeight/data/tree/master/mayweather-mcgregor)

A note about autograding

Because there are many ways to generate the correct visualization, we will not be using the autograder for this assignment. Compare your result to the provided samples. Try to get as close to our provided solution as possible.

Part 1. Data Types & Design (30 points)

Read the article published in Five Thirty Eight The Mayweather-McGregor Fight, As Told Through Emojis (https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/) and answer the following questions:

1.1 List the different data types in the following visualizations and their encodings (10 points)

For each chart, describe the variable name, type, and encoding (e.g., weight, quantitative, bar length)
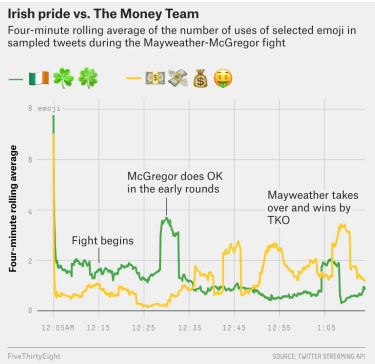
For the first chart:

1. Name: Percent; Type: Quantitative; Encoding: Bar length and List
2. Name: Emoji; Type: Nominal; Encoding: List

For the second chart:

1. Name: Emoji; Type: Nominal; Encoding: Color;
2. Name: Average frequency of occurences; Type: Quantitative; Encoding: Line height
3. Name: Time; Type: Time; Encoding: Time series

1.2 Sketch a visualization with an alternative encoding for one of the charts above. Compare your solution to the original. (10 points)

You can hand sketch or create the solution digitally. Please upload an image or screenshot. Your data doesn't need to match perfectly. Reflect on the differences in terms of perception/cognition and design principles as appropriate.
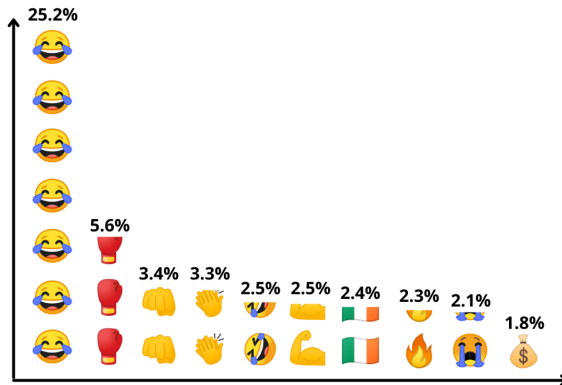
I don't use other kinds of chart for this question because I believe the bar chart is the most efficient way to compare items. I replace bar by emoji and make it more attractive and memorable. My vis contains percentage as well in order to not losing quantitive information.

1.3 Use one of the design principles reviewed in class (Tufte's data-ink ratio, graphical integrity, chart junk, etc.) to critique the following visualization (10 points)
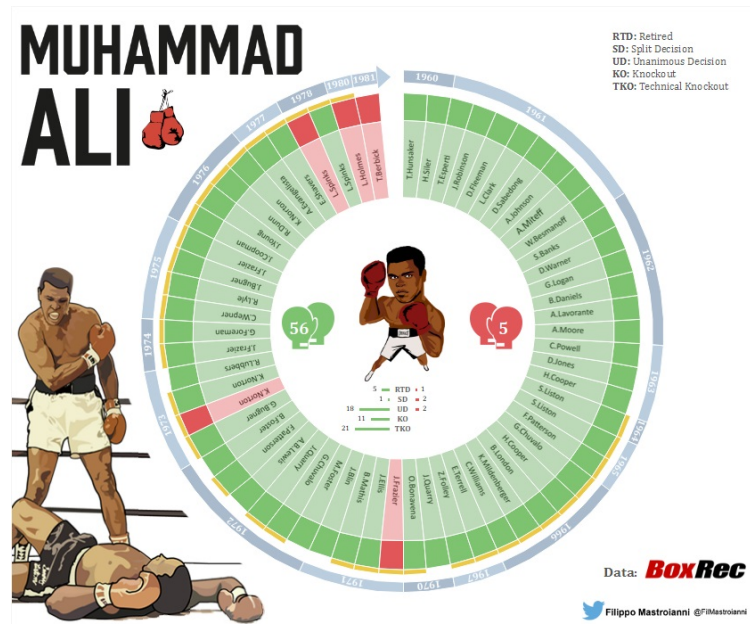
Describe pros and cons of the visualization relative to these principles. If the image violates the principle, reflect on why this is might be ok or not.

http://vizzingdata.blogspot.com/2017/01/muhammad-ali-career-dataviz.html

PRO:

1. This visualization contains almost all information in one chart. The audidence can figure out how many matches Ali wins and loses easily.
2. This visualization contains illustrations and makes more attractive and memorable than flat lists.

CON:

1. The proportion of size of texts and the chart is not balanced. Texts is too small to read and half of them are upside down.
2. This visualization looks like a pie chart, but it is difficult to interpret the proportion of the quantity of matches for each year.
3. This visualization lacks the information/explanation of yellow bars. It will confuse the audience who are not the expert of boxing matches. Not all readers have interests to go deeply through the interactive visualization and seek the meaning of yellow bars.
4. The original and the Blogspot's visualizations are not match each other. The original one contains the information of yellow bars, but loses the number of wins. The Blogspot's one doesn't have the information of yellow bars, but shows the number of wins. It might be occurred because of the layout on different platforms/softwares. Moreover, it is definitely caused by human factors, like the authors or editors' carelessness.

5. The ink ratio is probably large in view of blocks of green/red, between yellow bars and lightgreen/lightred blocks, and they doesn't contain any information or data.

Part 2. Altair programming exercise (70 points)

We have provided you with some code and parts of the article The Mayweather-McGregor Fight, As Told Through Emojis (https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/). This article is based on the dataset:

1. 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)

To earn points for this assignment, you must:

- Recreate the visualizations in the article (replace the images in the article with a code cell that creates a visualization). There are four visualizations to make. For the 2nd, 3rd, and 4th, we provide some example code to get the data into the right structure. The points for each visualization are distributed: (30 points: 9 (1st problem) + 7 (each of 2nd, 3rd & 4th)).
    - *Partial credit can be granted for each visualization (up to 5 points) if you provide the grammar of graphics description of the visualization without a fully implemented Altair solution*
- Propose one alternative visualization for one of the 4 article visualizations. Add a short paragraph describing why your visualization is better in terms of Design / Variable types encoding. (20 points/ 15 points plot + 5 justification)

- Propose a new visualization to complement a part of the article. Add a short paragraph justifying your decisions in terms of Design / Variable types encoding. (20 points/ 15 points plot + 5 justification)

Before you begin

IMPORTANT BROWSER ISSUE: For some non-ES6 Browsers there are problems with date/time conversions (see this (https://altair-viz.github.io/user_guide/times_and_dates.html)). If things aren't working try something like Chrome for this assignment.

IMPORTANT DATA ISSUE: There are some differences in the data that 538 used and what we have. This will cause some issues to how things are normalized. Things should look very similar, but you may find, for example, that the scale of tweets when you normalize is different than the original figure. This is fine.

IMPORTANT STYLING/ANNOTATION NOTE: Part of this assignment is to get styling and annotation to be as close to 538 as possible. Altair and Vega-Lite aren't super helpful for annotation purposes. You will find that you need to do things like layering text and the arrows (hint: see here (https://github.com/altair-viz/altair/issues/1721)). What I usually do in practice (when I need the visualization to look good and have annotation) is get things as close to how I want them to look, export the image as an SVG and then load it into Figma (https://www.figma.com/), InkScape (https://inkscape.org/), or Adobe Illustrator (https://www.adobe.com/products/illustrator.html). You can see "reasonably close approximations" in the examples we provide. Those have been generated using nothing but Altair.

```
In [2]:  # start with the setup
         import pandas as pd
         import altair as alt
         import numpy as np
```

```
In [3]:  # enable correct rendering
         alt.renderers.enable('default')
```

```
Out[3]:  RendererRegistry.enable('default')
```

```
In [4]:  # uses intermediate json files to speed things up
         alt.data_transformers.enable('json')
```

```
Out[4]:  DataTransformerRegistry.enable('json')
```

```
In [5]:   # we're going to do some setup here in anticipation of needing the data in
          # a specific format. We moved it all up here so everything is in one place.

          # 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 create a new column
          # that counts them
          tweets['emojis'] = tweets['text'].str.findall(r'[^\w\s.,"@\'?/#!$%\^&\*;:{}=\-_`~()\U0001F1E6-\U0001F1FF]').str.l

          # next, there are a few specific emojis that we care about, we're going to create
          # a column for each one and indicate how many times it showed up in the tweet
          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['💰'] +  tweets['💱']

In [6]:   # uncomment to see what's inside
          # tweets.head()
```

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]

> 1. We used the Twitter Streaming API (https://dev.twitter.com/streaming/overview) which provides a sample of all the tweets that matched our search terms: #MayMac, #MayweatherMcGregor, #MayweatherVMcGregor, #MayweatherVsMcGregor, #McGregor and #Mayweather. Of the 197,989 tweets we collected between 12:05 a.m. and 1:15 a.m. EDT, 12,118 had emojis.

	EMOJI	PERCENT	
1	😂	25.2%	████████████
2	👊	5.6	█
3	🗯	3.4	█
4	👏	3.3	█
5	🤑	2.5	█
6	💪	2.5	█
7	IE	2.4	█
8	🔥	2.3	█
9	😭	2.1	█
10	💰	1.8	█

The most-used emoji in over 240,000 tweets collected
during the Mayweather-McGregor fight broadcast,
from 12:05 a.m. to 1:30 a.m. EDT on Aug. 27.

** Homework note, construct your solution to this chart in the cell below. Click here (assets/altair_chart1.png) to see a sample output from Altair.

In [7]:
```python
# We'll help you out with a table that has the percentages for each emoji

# 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 = ['😂','🤑','🗯','👏','🔥','IE','💪','🔥','👊','💰']
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).reset_index()

# rename the columns
percentages_df = percentages_df.rename(columns={'index':'EMOJI', 0: 'PERCENT'})

# 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 [8]:
```python
# uncomment to see what's inside
# percentages_df
```

** Homework note, construct your solution to this chart in the cell below. Click here (assets/emoji_distrib_altair.png) to see a sample output from Altair.

In [9]:
```python
name_sort = ['😂','🗯','👏','👊','💪','IE','🤑','🔥','👊','💰']
```

```
In [10]: bars = alt.Chart(percentages_df).mark_bar(color='orange').encode(
             alt.X('PERCENT:Q',axis=None),
             alt.Y('EMOJI:N',sort='-x',title=None, axis=None)
         )
         bars
```

Out[10]:
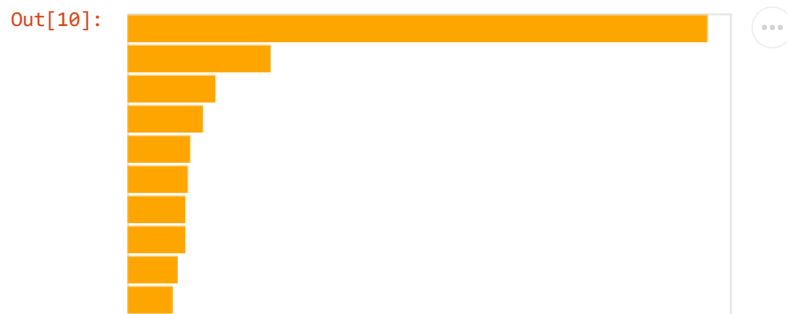

```
In [11]: emoji = alt.Chart(percentages_df).mark_bar().encode(
             y=alt.Y('EMOJI:N', axis=None, sort=name_sort),
             text=alt.Text('EMOJI:N')
         ).mark_text()
         emoji
```

Out[11]:


```
In [12]: text = alt.Chart(percentages_df).mark_bar().encode(
             y=alt.Y('EMOJI:N', axis=None, sort=name_sort),
             text=alt.Text('PERCENT_TEXT:N')
         ).mark_text()
         text
```

Out[12]:

```
23.1 %
 5.7 %
 3.5 %
 3.0 %
 2.5 %
 2.4 %
 2.3 %
 2.3 %
 2.0 %
 1.8 %
```
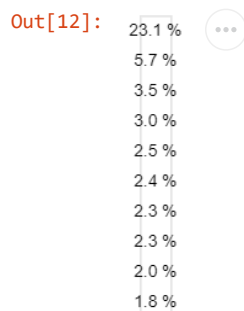
```
alt.hconcat(emoji,text,bars,spacing=10).configure(
    # customize background color
    background="whitesmoke"
).configure_view(
    strokeWidth=0
).configure_axis(
    # remove axis line
    grid=False, domain=False, ticks=False
)
```

😂	23.1 %
🥊	5.7 %
🐔	3.5 %
👏	3.0 %
💪	2.5 %
IE	2.4 %
🥴	2.3 %
🔥	2.3 %
🥶	2.0 %
🏅	1.8 %
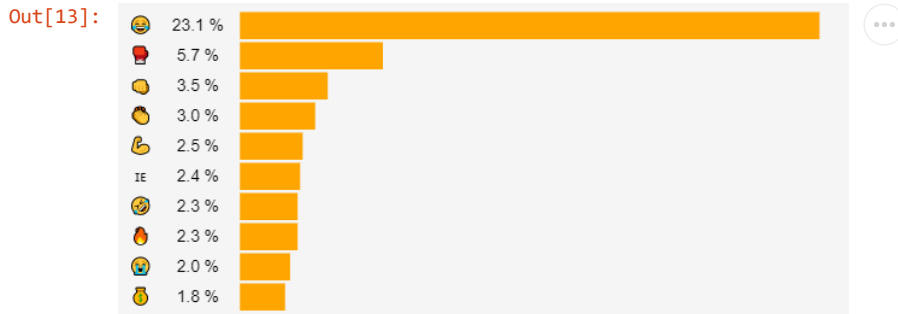
2.1 use percentages_df to recreate the visualization above

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.)

> **Nick**
> @Evil_Empire_44
>
> Fight time 🥊🥊 #MayweathervMcgregor
> ♡ 12:05 AM - Aug 27, 2017
>
> See Nick's other Tweets >

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.

Irish pride vs. The Money Team
Four-minute rolling average of the number of uses of selected emoji in sampled tweets during the Mayweather-McGregor fight

---
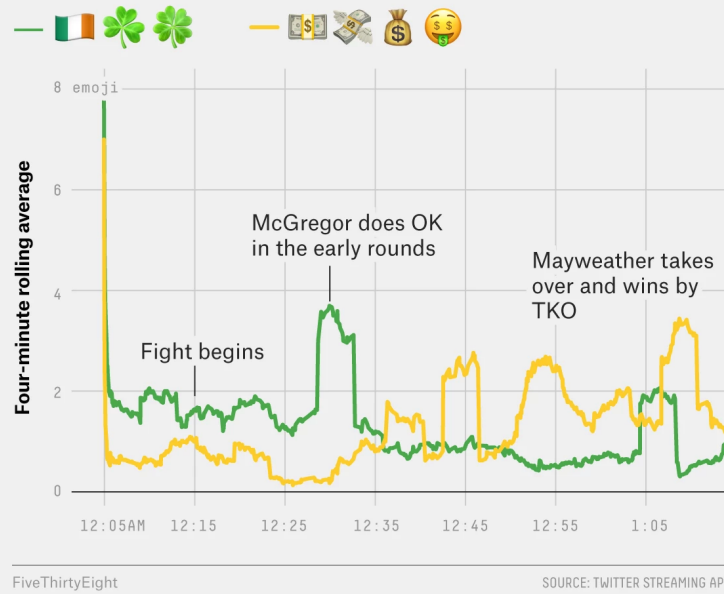
In [14]:
```python
# Again, we're going to help you set up the data

# We're going to want to work with time objects so we need to make a datetime
# column (basically transforming the text in "created at"). It duplicates
# the data but it will make things easier


tweets['datetime'] = pd.to_datetime(tweets['created_at'])
tweets = tweets.set_index('datetime')


teams = tweets.copy()
teams['irish_pride']
teams = teams.resample('1s').sum()
teams  = teams[(teams['💸']>0) | (teams['🤑']>0) | (teams['💰']>0) | (teams['🔢']>0) | (teams['💮']>0) | (teams

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

# next for the irish team
idf = teams['irish_pride'].rolling('4Min').mean().reset_index()
idf['team'] = '💮💮IE'
idf = idf.rename(columns={'irish_pride':'tweet_count'})

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

In [15]:
```python
# we're also going to create an annotations data frame to help you
annotations = [['2017-08-27 00:15:00',4, 'Fight begins'],
               ['2017-08-27 00:22:00',5.5, 'McGregor does OK \nin the early rounds'],
               ['2017-08-27 00:53:00',4.5, 'Mayweather takes \nover and wins by \nTKO']]
a_df = pd.DataFrame(annotations, columns=['date','count','note'])
```

** Homework note, construct your solution to this chart in the cell below. Click here (assets/altair_chart2.png) to see a sample output from Altair.

```
In [16]: # your turn, create your solution
         # YOUR CODE HERE
         achart = alt.Chart(ndf).mark_line().encode(
             alt.X('datetime:T', title=None,axis=alt.Axis(tickCount=8)),
             alt.Y('tweet_count:Q', title='Four-minute rolling average'),
             color=alt.Color('team:N', legend=alt.Legend(orient="top", title=None, symbolSize=20, labelFontSize=20), scale
         )
         ).properties(
             width=400,
             height=200,
             title={
                 'text':['Irish Pride VS The Money Team'],
                 'subtitle':["Four-minute rolling average of the number of uses of selected emoji in","sampled tweets dur:
             }
         ).transform_fold(
             ['🍀🍀IE','💸🤑💰💵']
         )
         # raise NotImplementedError()
         achart
```

Out[16]:

Irish Pride VS The Money Team
Four-minute rolling average of the number of uses of selected emoji in
sampled tweets during the Mayweather-McGregor fight
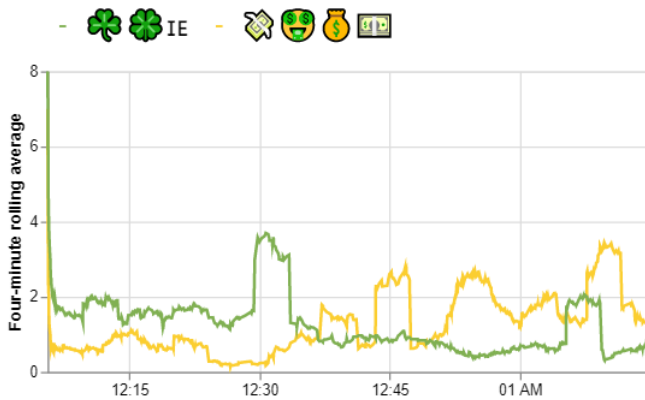

```
In [17]: a_annotation = alt.Chart(a_df).mark_text(lineBreak='\n').encode(
             x='date:T',
             y='count:Q',
             text='note:N'
         )

         aline1 = [['2017-08-27 00:15:00', 3.7],['2017-08-27 00:15:00', 2.1]]
         aline1_df=pd.DataFrame(aline1, columns=['date','count'])

         aline1 = alt.Chart(aline1_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )

         aline2 = [['2017-08-27 00:24:00', 4.6],['2017-08-27 00:30:00', 3.8]]
         aline2_df=pd.DataFrame(aline2, columns=['date','count'])

         aline2 = alt.Chart(aline2_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )
```
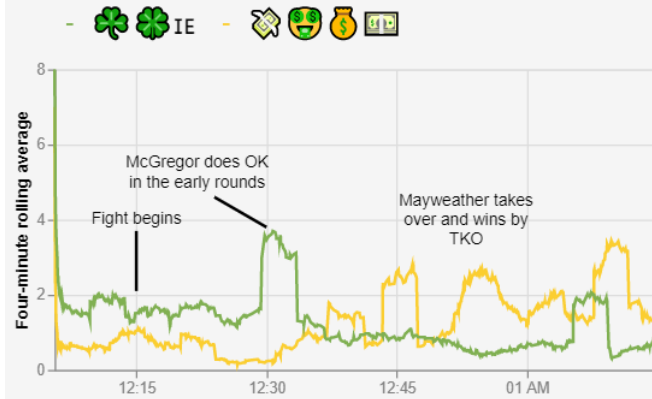
```
In [18]: alt.layer(achart, a_annotation, aline1, aline2).configure(
             # customize background color
             background="whitesmoke"
         ).configure_title(
             align='left',anchor='start',fontSize=20,fontWeight=700,subtitleFontSize=16
         ).configure_axis(
             labelColor='grey'
         )
```

Out[18]:

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.

Much hype, some boredom

Four-minute rolling average of the number of uses of selected emoji in sampled tweets during the Mayweather-McGregor fight

FiveThirtyEight

SOURCE: TWITTER STREAMING API
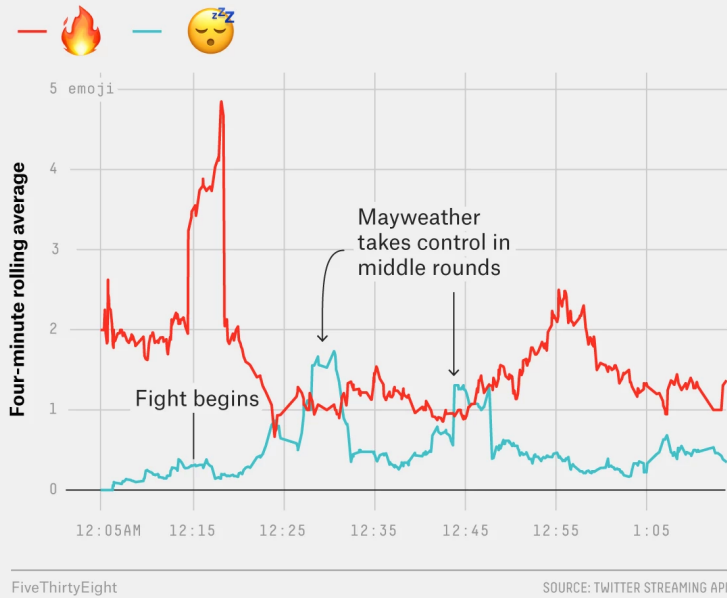
** Homework note, construct your solution to this chart in the cell below. Click here (assets/altair_chart3.png) to see a sample output from Altair.

2.3 your solution goes here, use the example above for the sampling and annotation

```
In [19]:  tweets['irish_pride'] = tweets['🔥']
          tweets['money_team'] = tweets['😴']
          # tweets.head()
```

```
In [20]:  tweets['datetime'] = pd.to_datetime(tweets['created_at'])
          tweets = tweets.set_index('datetime')


          teams = tweets.copy()
          teams['irish_pride']
          teams = teams.resample('1s').sum()
          teams  = teams[(teams['🔥']>0) | (teams['😴']>0)]


          sdf = teams['money_team'].rolling('4Min').mean().reset_index()
          sdf['team'] = '😴'
          sdf = sdf.rename(columns={'money_team':'tweet_count'})

          fdf = teams['irish_pride'].rolling('4Min').mean().reset_index()
          fdf['team'] = '🔥'
          fdf = fdf.rename(columns={'irish_pride':'tweet_count'})

          m_df = pd.concat([fdf,sdf])
```

```
In [21]: bchart = alt.Chart(m_df).mark_line().encode(
             alt.X('datetime:T', title=None,axis=alt.Axis(tickCount=8)),
             alt.Y('tweet_count:Q', title='Four-minute rolling average'),
             color=alt.Color('team:N', legend=alt.Legend(orient="top", title=None, symbolSize=20, labelFontSize=20), scale
         )
         ).properties(
             width=400,
             height=200,
             title={
                 'text':['Much hype, some boredom'],
                 'subtitle':["Four-minute rolling average of the number of uses of selected emoji in","sampled tweets duri
             }
         ).transform_fold(
             ['🔥','😵']
         )
         bchart
```

Out[21]:

Much hype, some boredom
Four-minute rolling average of the number of uses of selected emoji in
sampled tweets during the Mayweather-McGregor fight
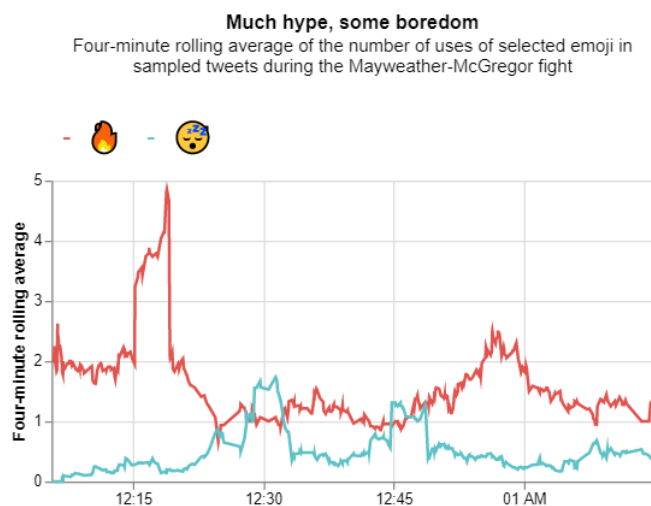
```
In [22]:  annotation2 = [['2017-08-27 00:15:00',1.3, 'Fight begins'],
                         ['2017-08-27 00:52:00',3, 'Mayweather takes control in middle rounds']]
          b_df = pd.DataFrame(annotation2, columns=['date','count','note'])

          b_annotation = alt.Chart(b_df).mark_text().encode(
              x='date:T',
              y='count:Q',
              text='note:N'
          )

          bline1 = [['2017-08-27 00:15:00', 1.1],['2017-08-27 00:15:00', 0.5]]
          c_df=pd.DataFrame(bline1, columns=['date','count'])
          line1 = alt.Chart(c_df).mark_line().encode(
              x='date:T',
              y='count:Q',
              color=alt.value("black")
          )

          bline2 = [['2017-08-27 00:33:00', 2.8],['2017-08-27 00:31:00', 1.9]]
          d_df=pd.DataFrame(bline2, columns=['date','count'])
          line2 = alt.Chart(d_df).mark_line().encode(
              x='date:T',
              y='count:Q',
              color=alt.value("black")
          )

          bline3 = [['2017-08-27 00:45:00', 2.7],['2017-08-27 00:45:00', 1.5]]
          e_df=pd.DataFrame(bline3, columns=['date','count'])
          line3 = alt.Chart(e_df).mark_line().encode(
              x='date:T',
              y='count:Q',
              color=alt.value("black")
          )
```
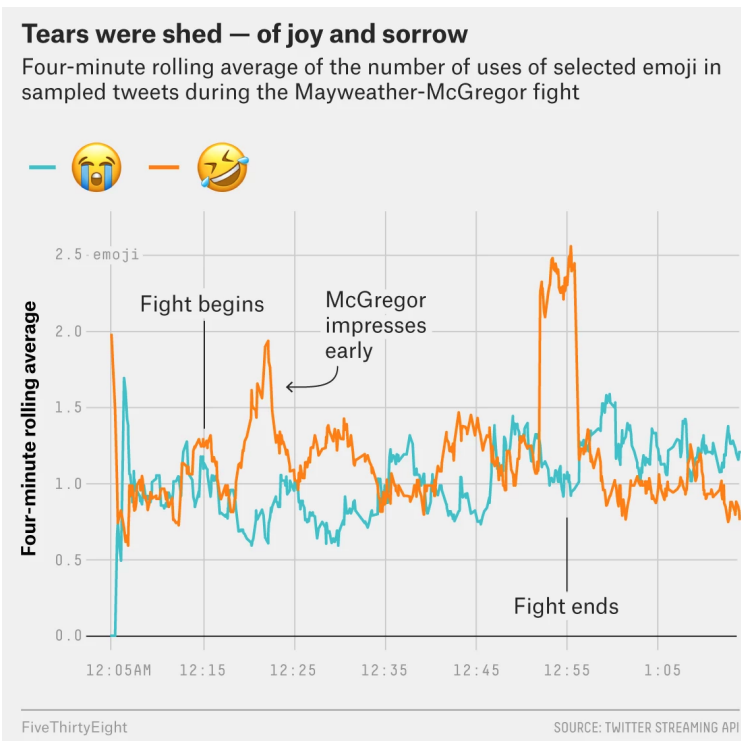
```
In [23]:  alt.layer(bchart, b_annotation, line1, line2, line3).configure(
              # customize background color
              background="whitesmoke"
          ).configure_title(
              align='left',anchor='start',fontSize=20,fontWeight=700,subtitleFontSize=16
          ).configure_axis(
              labelColor='grey'
          ).configure_view(
              strokeWidth=0
          )
```

Out[23]:

Much hype, some boredom
Four-minute rolling average of the number of uses of selected emoji in sampled tweets during the Mayweather-McGregor fight
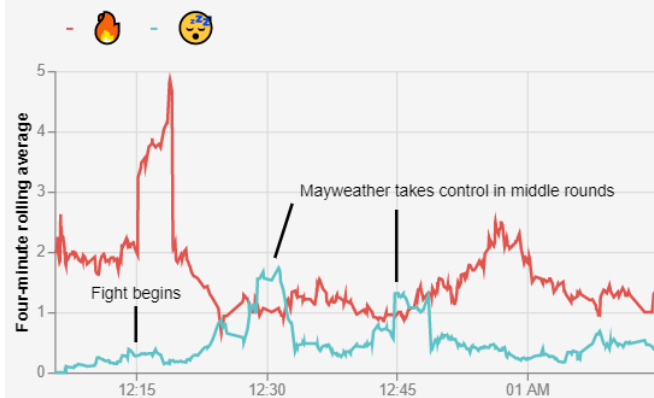
It ended just over 37 minutes after it began. Five seconds later, Mayweather leapt up on the corner ropes, victorious — 50-0 (https://fivethirtyeight.com/features/mayweather-is-defined-by-the-zero-next-to-his-name/). Some observers declared it a satisfying spectacle (https://www.si.com/boxing/2017/08/27/after-months-hype-mayweather-and-mcgregor-deliver-boxing-spectacle). Others, McGregor chief among them, were frustrated with the finish (https://www.cbssports.com/boxing/news/conor-mcgregor-frustrated-with-refs-

. The emoji users on Twitter appeared to think the fight was, for the most part, 🔥 — especially as it heated up toward the end. While the result may never have been in question, this was a welcome outcome for many who viewed Mayweather's last megafight against Manny Pacquiao as an epic 😴😴😴😴.

K.Zoldik
@John__Alp

Yeeesssss !! 🔥 #Mayweather

♡ 12:51 AM - Aug 27, 2017   ⓘ

👤 See K.Zoldik's other Tweets   〉

Tears were shed — of joy and sorrow
Four-minute rolling average of the number of uses of selected emoji in sampled tweets during the Mayweather-McGregor fight

** Homework note, construct your solution to this chart in the cell below. Click here (assets/altair_chart4.png) to see a sample output from Altair.

2.4 your solution goes here, use the example above for the sampling and annotation

```
In [24]: tweets['irish_pride'] = tweets['😭']
         tweets['money_team'] = tweets['🤣']
```

```
In [25]: tweets['datetime'] = pd.to_datetime(tweets['created_at'])
         tweets = tweets.set_index('datetime')


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

         cdf = teams['money_team'].rolling('4Min').mean().reset_index()
         cdf['team'] = '🤣'
         cdf = cdf.rename(columns={'money_team':'tweet_count'})

         hdf = teams['irish_pride'].rolling('4Min').mean().reset_index()
         hdf['team'] = '😭'
         hdf = hdf.rename(columns={'irish_pride':'tweet_count'})

         odf = pd.concat([cdf,hdf])
```

```
In [26]: c_chart = alt.Chart(odf).mark_line().encode(
             alt.X('datetime:T', title=None,axis=alt.Axis(tickCount=8)),
             alt.Y('tweet_count:Q', title='Four-minute rolling average'),
             color=alt.Color('team:N', legend=alt.Legend(orient="top", title=None, symbolSize=20, labelFontSize=20), scale
         )
         ).properties(
             width=400,
             height=250,
             title={
                 'text':['Tears were shed-of joy and sorrow'],
                 'subtitle':["Four-minute rolling average of the number of uses of selected emoji in","sampled tweets dur:
             }
         ).transform_fold(
             ['😭','🤣']
         )
         c_chart
```

Out[26]:

Tears were shed-of joy and sorrow
Four-minute rolling average of the number of uses of selected emoji in
sampled tweets during the Mayweather-McGregor fight
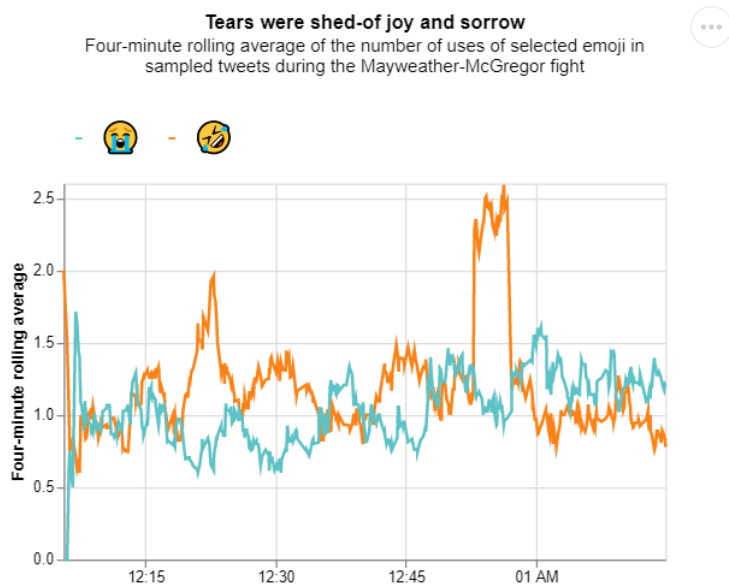
```
In [27]: annotation2 = [['2017-08-27 00:10:00',2.2, 'Fight begins'],
                        ['2017-08-27 00:30:00',2.0, 'McGregor\nimpresses\nnearly'],
                        ['2017-08-27 00:50:00',0.1,'Fight ends']]
         c_df = pd.DataFrame(annotation2, columns=['date','count','note'])

         c_annotation = alt.Chart(c_df).mark_text(lineBreak='\n',align='left').encode(
             x='date:T',
             y='count:Q',
             text='note:N'
         )
```

```
In [28]: cline1 = [['2017-08-27 00:15:00', 2.1],['2017-08-27 00:15:00', 1.4]]
         x_df=pd.DataFrame(cline1, columns=['date','count'])

         cline1 = alt.Chart(x_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )

         cline2 = [['2017-08-27 00:29:00', 1.8],['2017-08-27 00:24:00', 1.6]]
         y_df=pd.DataFrame(cline2, columns=['date','count'])

         cline2 = alt.Chart(y_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )

         cline3 = [['2017-08-27 00:55:00', 0.3],['2017-08-27 00:55:00', 0.8]]
         z_df=pd.DataFrame(cline3, columns=['date','count'])

         cline3 = alt.Chart(z_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )
```
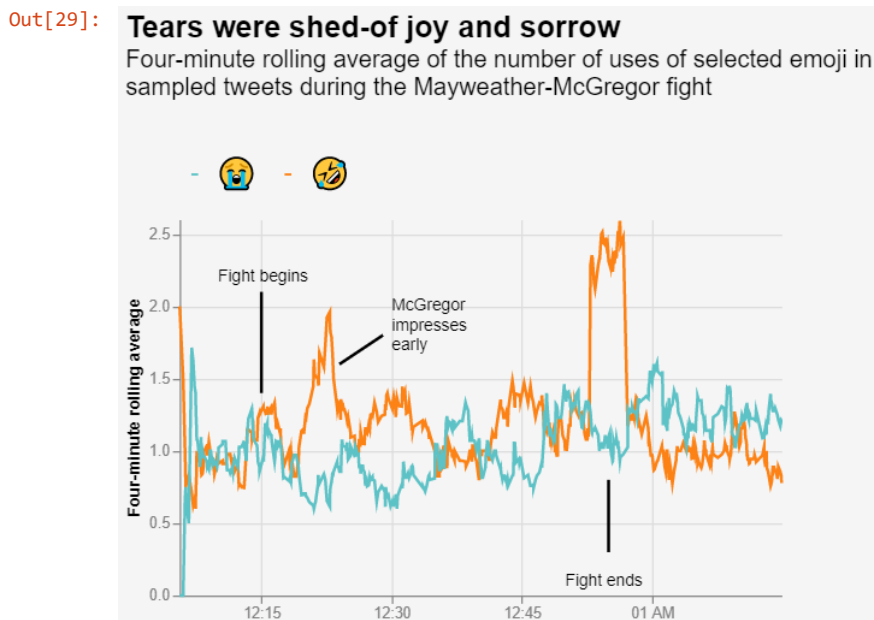
```
In [29]: alt.layer(c_chart, c_annotation, cline1, cline2, cline3).configure(
             # customize background color
             background="whitesmoke"
         ).configure_title(
             align='left',anchor='start',fontSize=20,fontWeight=700,subtitleFontSize=16
         ).configure_axis(
             labelColor='grey'
         ).configure_view(
             strokeWidth=0
         )
```

Out[29]:

Tears were shed-of joy and sorrow
Four-minute rolling average of the number of uses of selected emoji in
sampled tweets during the Mayweather-McGregor fight

They laughed. They cried. And they laughed some more. And they cried some more.

2.5 Make your own (part 1-alternative)

Propose one *alternative* visualization for one of the article's visualizations. Add a short paragraph describing why your visualization is more *effective* based on principles of perception/cognition. If you feel your visualization is worse, that's ok! Just tell us why. (20 points/ 15 points plot + 5 justification)

In [30]:
```python
my_chart1 = alt.Chart(odf).mark_area(opacity=0.3).encode(
    x=alt.X("datetime:T",title=None,axis=alt.Axis(tickCount=8)),
    y=alt.Y("tweet_count:Q", stack=None, title='Four-minute rolling average'),
    color=alt.Color("team:N", title='EMOJI',scale=alt.Scale(
            domain=['😭','🤣'],
            range=['#965454', '#8698a7']))
).properties(
    width=400,
    height=250,
    title={
        'text':['Tears were shed-of joy and sorrow'],
        'subtitle':["Four-minute rolling average of the number of uses of selected emoji in","sampled tweets duri
    }
)
my_chart1
```

Out[30]:

Tears were shed-of joy and sorrow
Four-minute rolling average of the number of uses of selected emoji in
sampled tweets during the Mayweather-McGregor fight
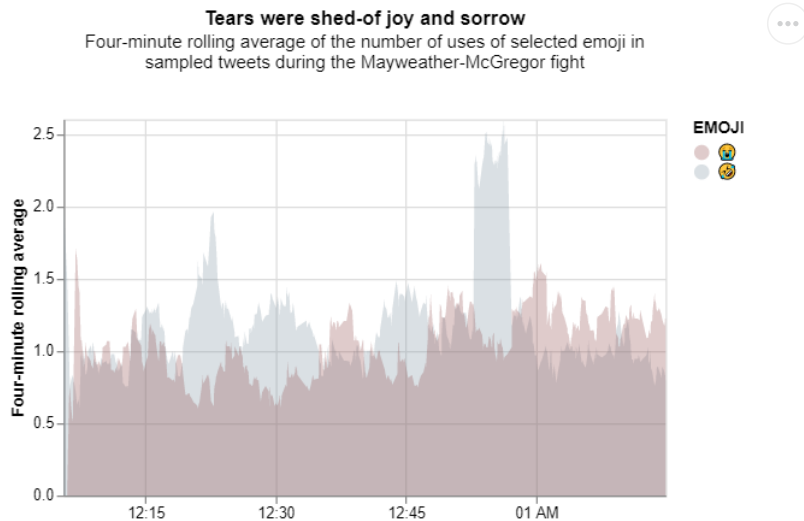
In [31]:
```python
alt.layer(my_chart1, c_annotation, cline1, cline2, cline3).configure_title(
    align='left',anchor='start',fontSize=20,fontWeight=700,subtitleFontSize=16
).configure_axis(
    labelColor='grey'
).configure_view(
    strokeWidth=0
)
```

Out[31]:

Tears were shed-of joy and sorrow
Four-minute rolling average of the number of uses of selected emoji in
sampled tweets during the Mayweather-McGregor fight
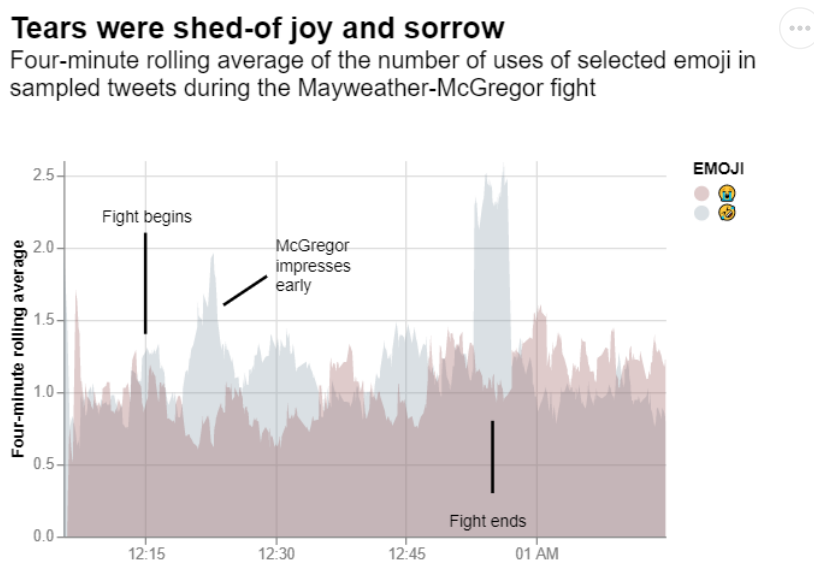
I created an area chart for the last visualization. I believe my chart and the 538 chart provide almost same functions to the audience. The only disadvantage of my chart is that a part of areas is covered by another one. Therefore, the 538's visualization is more effective than mine.

2.6 Make your own (part 2-novel)

Propose a *new* visualization to complement a part of the article. Add a short paragraph justifying your decisions in terms of Perception/Cognition processes. If you feel your visualization is worse, that's ok! Just tell us why. (20 points/ 15 points plot + 5 justification)

In [32]:
```python
# YOUR CODE HERE
# raise NotImplementedError()
tweets['irish_pride'] = tweets['👏']
tweets['money_team'] = tweets['🥊']
tweets['datetime'] = pd.to_datetime(tweets['created_at'])
tweets = tweets.set_index('datetime')


teams = tweets.copy()
teams['irish_pride']
teams = teams.resample('1s').sum()
teams  = teams[(teams['👏']>0) | (teams['🥊']>0)]

my_df1 = teams['money_team'].rolling('4Min').mean().reset_index()
my_df1['team'] = '🥊'
my_df1 = my_df1.rename(columns={'money_team':'tweet_count'})

my_df2 = teams['irish_pride'].rolling('4Min').mean().reset_index()
my_df2['team'] = '👏'
my_df2 = my_df2.rename(columns={'irish_pride':'tweet_count'})

my_df = pd.concat([my_df1,my_df2])
```

In [33]:
```python
my_chart = alt.Chart(my_df).mark_area(opacity=0.3).encode(
    x=alt.X('datetime:T', title=None,axis=alt.Axis(tickCount=8)),
    y=alt.Y('tweet_count:Q', stack=None, title='Four-minute rolling average'),
    color=alt.Color('team:N', title='EMOJI',scale=alt.Scale(
        domain=['👏','🥊'],
        range=['#965454', '#8698a7'])
)
).properties(
    width=400,
    height=250,
    title={
        'text':['Cheers when Mayweather took advantage', 'and won the match'],
        'subtitle':["Four-minute rolling average of the number of uses of selected emoji in","sampled tweets duri
    }
).transform_fold(
    ['👏','🥊']
)
my_chart
```

Out[33]:
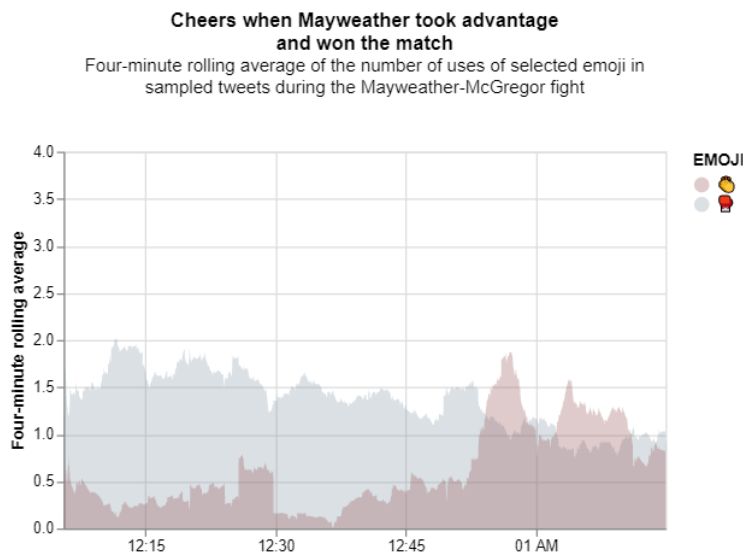
```
In [34]: annotation26 = [['2017-08-27 00:10:00',3.0, 'Fight begins'],
                         ['2017-08-27 00:23:00',2.5, 'McGregor winning \nrounds as aggressor'],
                         ['2017-08-27 00:50:00',0.1,'Fight ends']]
         annot26_df = pd.DataFrame(annotation26, columns=['date','count','note'])

         annot_26 = alt.Chart(annot26_df).mark_text(lineBreak='\n',align='left').encode(
             x='date:T',
             y='count:Q',
             text='note:N'
         )
```

```
In [35]: myline1 = [['2017-08-27 00:15:00', 2.8],['2017-08-27 00:15:00', 2.0]]
         my1_df=pd.DataFrame(myline1, columns=['date','count'])

         myline1 = alt.Chart(my1_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )

         myline2 = [['2017-08-27 00:30:00', 2.1],['2017-08-27 00:30:00', 1.6]]
         my2_df=pd.DataFrame(myline2, columns=['date','count'])

         myline2 = alt.Chart(my2_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )

         myline3 = [['2017-08-27 00:55:00', 0.3],['2017-08-27 00:55:00', 0.8]]
         my3_df=pd.DataFrame(myline3, columns=['date','count'])

         myline3 = alt.Chart(my3_df).mark_line().encode(
             x='date:T',
             y='count:Q',
             color=alt.value("black")
         )
```

```
In [36]: alt.layer(my_chart, annot_26, myline1, myline2, myline3).configure(
             # customize background color
             background="whitesmoke"
         ).configure_title(
             align='left',anchor='start',fontSize=20,fontWeight=700,subtitleFontSize=16
         ).configure_axis(
             labelColor='grey'
         ).configure_view(
             strokeWidth=0
         )
```

Out[36]:

Cheers when Mayweather took advantage and won the match
Four-minute rolling average of the number of uses of selected emoji in sampled tweets during the Mayweather-McGregor fight
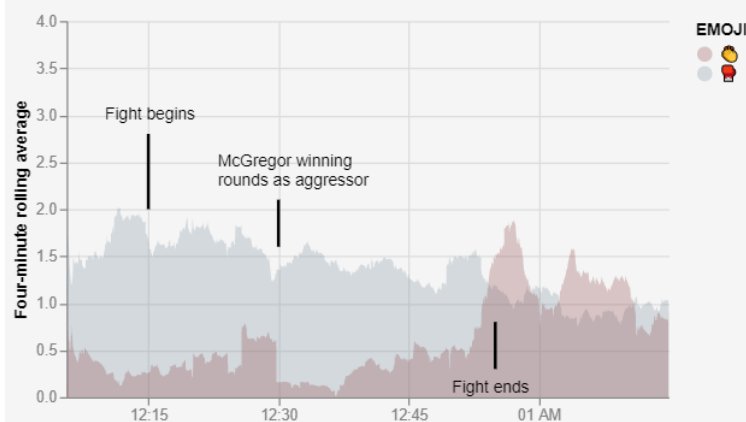
Based on the chart, we can see that even though the majority of audiences are big fans of McGregor, they acclaimed on Tweeters when Mayweather took advantage during the 3rd round and got the UFC Lightweight Champion finally.

I use the area chart visualization comparing "cheers" and "fight" emojis on Tweeters. We, the readers, interpret that the audiences' attitude to the result of the match and they showed a great of respect to McGregor, even though most of them are fans of Mayweather.

In [ ]: