

Information Visualization I

School of Information, University of Michigan

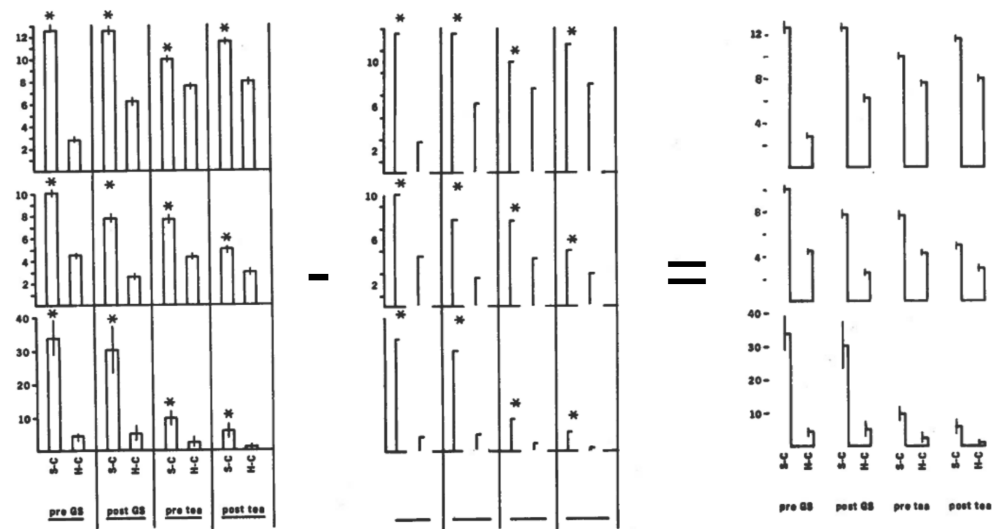
Week 4:

- 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/) (<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) (<https://fivethirtyeight.com>) available [online](https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/) (<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) (<https://github.com/fivethirtyeight/data/tree/master/mayweather-mcgregor>)

Important notes:

- 1) Grading for this assignment is entirely done by manual inspection. 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.
- 2) Depending on your operating system and browser combination your emojis may not exactly match ours or the ones from 538. That's fine.
- 3) When turning in your PDF, please use the File -> Print -> Save as PDF option **from your browser**. Do **not** use the File->Download as->PDF option. Complete instructions for this are under Resources in the Coursera page for this class.

Part 1. Data Types & Design (30 points)

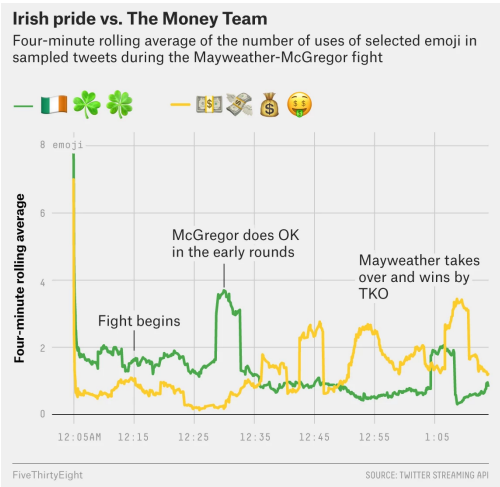
Read the article published in Five Thirty Eight [The Mayweather-McGregor Fight, As Told Through Emojis](https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/) (<https://fivethirtyeight.com/features/the-mayweather-mcgregor-fight-as-told-through-emojis/>) and answer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

	EMOJI	PERCENT	
1	😓	25.2%	<div></div>
2	👊	5.6	<div></div>
3	👊	3.4	<div></div>
4	👊	3.3	<div></div>
5	👊	2.5	<div></div>
6	👊	2.5	<div></div>
7	IE	2.4	<div></div>
8	🔥	2.3	<div></div>
9	😭	2.1	<div></div>
10	💰	1.8	<div></div>

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.

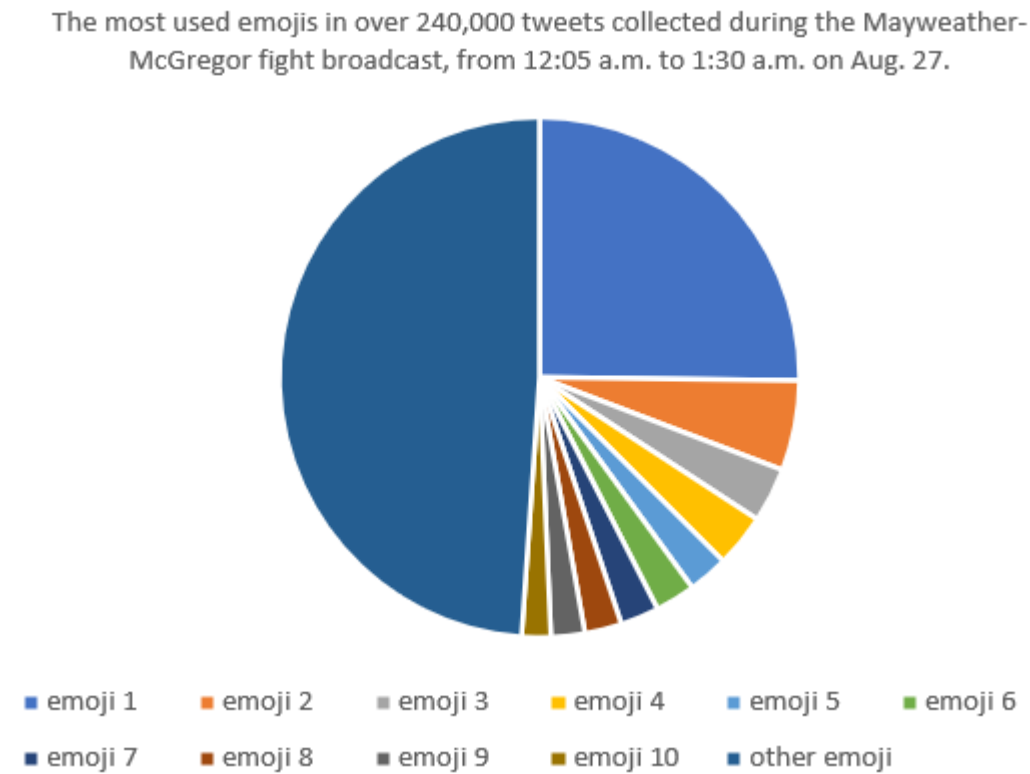


For the first chart, the emojis used in the tweets during the Mayweather-McGregor fight are a nominal variable that is encoded by position (its location on the left portion of the chart). The emoji's use percentage relative to other emojis is a quantitative variable that is encoded by bar length as well as text itself, which allows for an effective display as length alone would have been difficult to interpret as many of the lengths look similar.

The second chart has encoded time-series data on the x-axis. In this context, the time is an ordinal variable and will be used to track the usage of tweets at various portions of the night. The 4-minute rolling average of tweets containing various emojis was encoded on the y-axis and is a quantitative variable. The value itself is encoded by position within the (x,y) plane. While rolling average was encoded in the y position and time in the x position, emoji groups corresponding to Mayweather and McGregor are nominal variables that were encoded by color. With these encodings together, the viewer is able to track 4-minute rolling averages over time throughout the night for various emojis of interest.

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

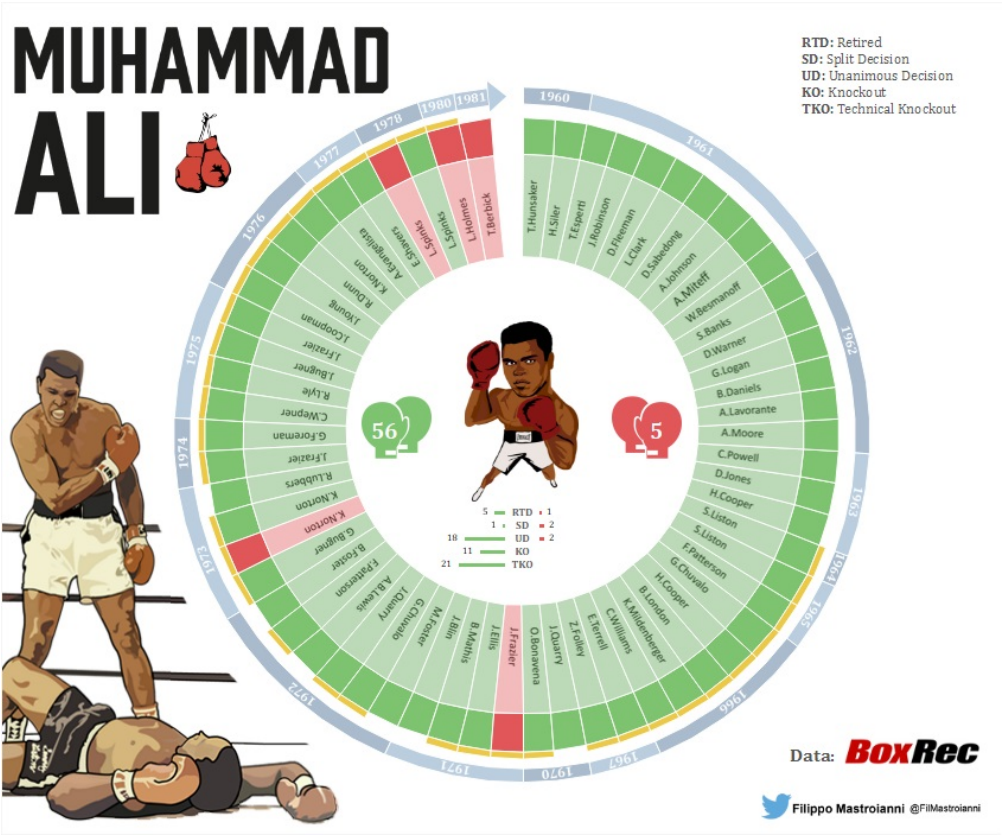
You can hand sketch or create the solution digitally. Please upload an image or screenshot. Your data doesn't need to match perfectly. Reflect on the differences in terms of perception/cognition and design principles as appropriate.



For simplicity in generating the alternative encoding, emoji 1 corresponds to the laughing emoji (index 1 above), emoji 2 corresponds to the boxing glove (index 2 above), etc. This is an alternative encoding that has the same level of expressiveness as the first chart above as it shows the relative usage of emojis when compared to the top 10 emojis. An "other" category was added in to complete the pie but the same information is therefore found in this figure as the chart above. However, the chart from the article is far more effective as the percentages give a clear breakout of each emoji's usage. Length is also much easier for a viewer to decipher, especially when compared to a pie chart where some slices are very similar to others, thus making the percentages difficult to discern. Additionally, this is a good example of Munzner's Rule of thumb that if a list is good enough, a 2D image is not justified. The list above with the emoji and percentages (even excluding the bar chart) is a very effective chart that would not need to be displayed to get its overall point across.

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>

The above figure violates the principle of chart junk. Tufte, a minimalist in terms of preference around visualization and display, had a rule that "thou shalt not use chart junk". Chart junk refers to images or displays on a chart that are essentially irrelevant to the data itself and instead play a role in calling attention to the figure. However, there are pros and cons to this approach and I believe that overall the use of chart junk here is effective. As a con, on the far left is an extremely iconic picture of Muhammad Ali, which could serve to distract the reader and even result in a reader just viewing the picture and continuing to scroll look past the image without digesting/reflecting on it. These types of images and pictures outside of the chart serve to draw a reader's eyes away from the overall point of the display.

However, as a pro, the reader is also drawn to the miniature picture of Ali as well as the green and red boxing gloves denoting his overall record. A benefit of chart junk displays is that while there is a lot more going on in the chart itself, the reading is still able to have very similar if not exactly comprehension of the figure relative to if the junk didn't exist at all. For example, it's clear that the circle itself is an ordinal variable tracking chronologically Ali's fights and that the names of all his opponents are along the wheel, with green denoting wins and red losses. The graphics in the middle of the circle, also are a great summary of the overall record and how the wins and losses were achieved. So, while there is chart junk, the

data is displayed in a way that is easily digestible. Additionally, a pro of chart junk figures is how they lure the reader's eyes in and how memorable they are. I believe this chart is no exception and I feel as though it will be easy to walk away from viewing this figure and recalling that Ali had a 56-5 record and can perhaps even remember which boxers he lost to. So, while there is a lot of chart junk in this visual, I believe it benefits from this overall in terms of how memorable it makes it.

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/) (<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) (<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) (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) (<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/) (<https://www.figma.com/>), [InkScape](https://inkscape.org/) (<https://inkscape.org/>), or [Adobe Illustrator](https://www.adobe.com/products/illustrator.html) (<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 [1]: # start with the setup
```

```
import pandas as pd
import altair as alt
import numpy as np
```

```
In [2]: # enable correct rendering
```

```
alt.renderers.enable('default')
```

```
Out[2]: RendererRegistry.enable('default')
```

```
In [3]: # uses intermediate json files to speed things up
```

```
alt.data_transformers.enable('json')
```

```
Out[3]: DataTransformerRegistry.enable('json')
```

```
In [4]: # 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.len()
```

```
# 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 = ['🍀', 'IE', '🍀', '🍀', '🍀', '🍀', '🍀', '🍀', '🍀', '🍀', '🍀', '🍀', '🍀', 'IE', '🍀', '🍀', '🍀', '🍀']
```

```
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 🍀, IE or 🍀 and 🍀, 🍀, 🍀 or 🍀 for each
```

```
tweets['irish_pride'] = tweets['🍀'] + tweets['IE'] + tweets['🍀']
```

```
tweets['money_team'] = tweets['🍀'] + tweets['🍀'] + tweets['🍀'] + tweets['🍀']
```

```
In [5]: # uncomment to see what's inside
tweets.head()
```

Out[5]:

	created_at	emojis	id	link	retweeted	screen_name	text		IE		...						
0	2017-08-27 00:05:34	1	901656910939770881	https://twitter.com/statuses/901656910939770881	False	aaLiysr	Ringe çıkmadan ateş etmeye başladı 😊 #McGregor ...	0	0	0	...	0	0	0	0	0	0
1	2017-08-27 00:05:35	5	901656917281574912	https://twitter.com/statuses/901656917281574912	False	zulmafrancozaf	🤔🤔🤔🤔🤔 @lalylourbet2 https://t.co/ERUGHhQINE	0	0	0	...	0	0	0	0	0	0
2	2017-08-27 00:05:35	2	901656917105369088	https://twitter.com/statuses/901656917105369088	False	Adriana11D	IEIEIE 🤔🤔 #MayweathervMcgregor	0	3	0	...	0	0	0	0	0	2
3	2017-08-27 00:05:35	2	901656917747142657	https://twitter.com/statuses/901656917747142657	False	Nathan_Caro_	Cest partit #MayweatherMcGregor 🤔	0	0	0	...	0	0	0	0	0	1
4	2017-08-27 00:05:35	2	901656916828594177	https://twitter.com/statuses/901656916828594177	False	sahouraxox	Low key feeling bad for ppl who payed to watch...	0	0	0	...	0	2	0	0	0	0

5 rows × 25 columns

The Mayweather-McGregor Fight, As Told Through Emojis

We laughed, cried and cried some more.

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

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

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

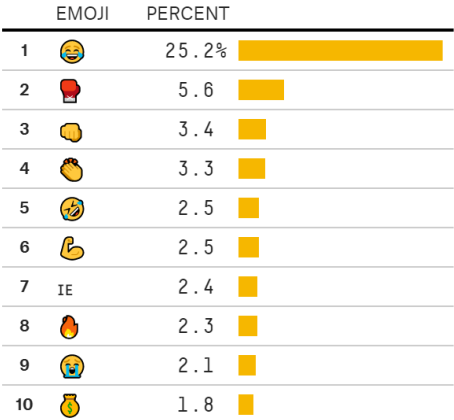
Get the data on [GitHub \(https://github.com/fivethirtyeight/data/tree/master/mayweather-mcgregor\)](https://github.com/fivethirtyeight/data/tree/master/mayweather-mcgregor)

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 \(http://www.espn.com/boxing/story/_/id/20469815/floyd-mayweather-conor-mcgregor-delay-ppv-problems\)](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 emojianalysis might not make it through peer review.)¹

1. We used the [Twitter Streaming API \(https://dev.twitter.com/streaming/overview\)](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.



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 [6]: # 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 emojis
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).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 [7]: # uncomment to see what's inside
percentages_df
```

Out[7]:

	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	IE	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 %

** 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.

2.1 use percentages_df to recreate the visualization above

```

In [8]: # use percentages_df to recreate the visualization above
alt.themes.enable('fivethirtyeight')

sorted_emoji = list(percentages_df.sort_values(by = 'PERCENT', ascending = False)['EMOJI'])
percentages_df['concat'] = percentages_df['EMOJI'] + ' ' + percentages_df['PERCENT_TEXT']

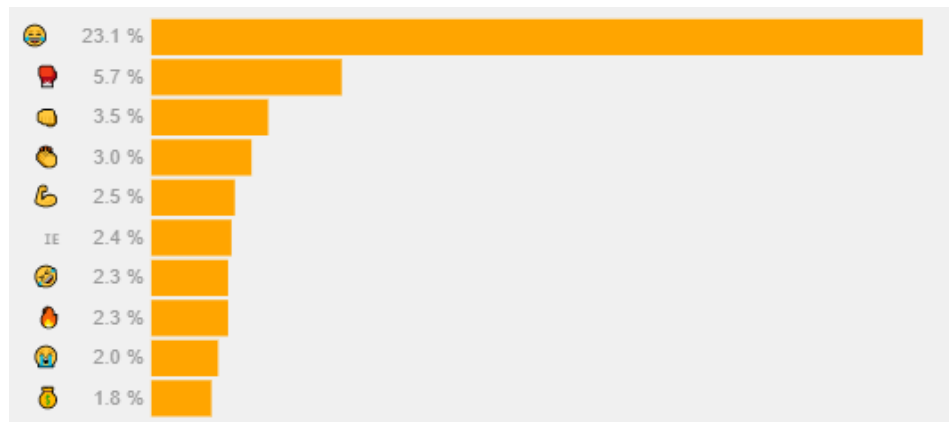
vis_21 = alt.Chart(percentages_df).mark_bar().encode(
    y=alt.Y('concat:N', axis=alt.Axis(domainOpacity=0, ticks=False), sort = sorted_emoji, title = None),
    x = alt.X('PERCENT:Q', axis = None),
    color = alt.value('orange'),
)

vis_21.configure_view(
    strokeOpacity=0
)

#raise NotImplementedError()

```

Out[8]:



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 \(http://emojitracker.com/\)](http://emojitracker.com/) emoji generally.

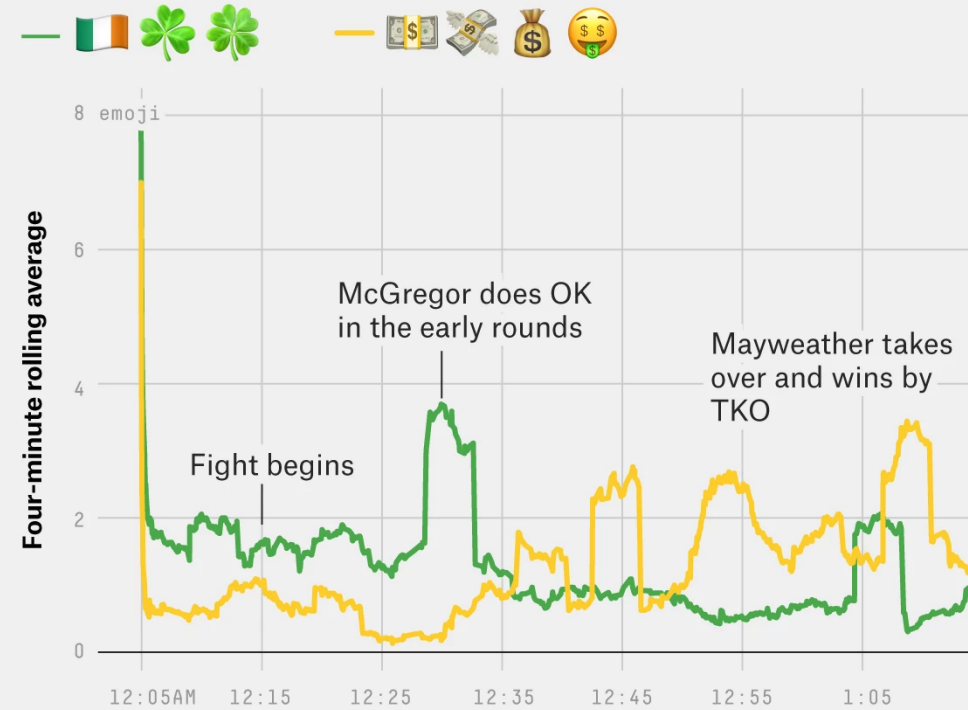
Here’s how the night unfolded, emoji-wise. (All of the charts below show them on a four-minute rolling average.)



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.

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 [9]: *# 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['🌸']>0) | (teams['IE']>0) ]
```

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 [10]: # uncomment to see what's inside
ndf.sample(5)
```

Out[10]:

	datetime	tweet_count	team
32	2017-08-27 00:06:38	0.636364	🇺🇸🇵🇪🇸🇩🇪
133	2017-08-27 00:11:57	1.812500	🇵🇪🇸🇩🇪IE
363	2017-08-27 00:34:09	0.840000	🇺🇸🇵🇪🇸🇩🇪
758	2017-08-27 01:10:02	3.340426	🇺🇸🇵🇪🇸🇩🇪
723	2017-08-27 01:07:21	1.352941	🇺🇸🇵🇪🇸🇩🇪

```
In [11]: # 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, 'McGregor does OK \nin the early rounds'],
               ['2017-08-27 00:53:00',4, 'Mayweather takes \nover and wins by \nTKO']]
a_df = pd.DataFrame(annotations, columns=['date', 'count', 'note'])
```

```
In [12]: # uncomment to see what's inside
a_df
```

Out[12]:

	date	count	note
0	2017-08-27 00:15:00	4	Fight begins
1	2017-08-27 00:22:00	5	McGregor does OK \nin the early rounds
2	2017-08-27 00:53:00	4	Mayweather takes \nover and wins by \nTKO

** 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.

2.2 your turn, create your solution


```

In [13]: # your turn, create your solution
vis_22 = alt.Chart(ndf).mark_line().encode(
    y=alt.Y('tweet_count:Q', title = 'Four-minute rolling average'),
    x=alt.X('datetime:T', axis=alt.Axis(format='%I:%M', tickCount = 4)),
    color = alt.Color('team', legend = alt.Legend(orient = 'top', symbolType = 'stroke'), title = None,
        scale = alt.Scale(range = ['green', '#FFD700']))
).properties(
    title={
        "text": ["Irish pride vs. The Money Team"],
        "subtitle": ["Four-minute rolling average of the number of uses of selected emoji in",
            "sampled tweets during the Mayweather-McGregor fight"]
    }
)

ann_22 = alt.Chart(a_df).mark_text(
    lineBreak = '\n').encode(
        y=alt.Y('count:Q'),
        x=alt.X('date:T', axis=alt.Axis(format='%I:%M', tickCount = 4, title = None)),
        text = 'note:N'
    )

lines_22 = [
    ['2017-08-27 00:15:00', 2.2, 'A'],
    ['2017-08-27 00:15:00', 3.8, 'A'],
    ['2017-08-27 00:26:00', 4.4, 'B'],
    ['2017-08-27 00:30:00', 3.9, 'B']]

l_df = pd.DataFrame(lines_22, columns=['x', 'y', 'group'])

l_22 = alt.Chart(l_df).mark_line().encode(
    y=alt.Y('y:Q'),
    x=alt.X('x:T', axis=alt.Axis(format='%I:%M', tickCount = 4, title = None)),
    detail = 'group',
    color = alt.value('black')
)

vis_22 + ann_22 + l_22

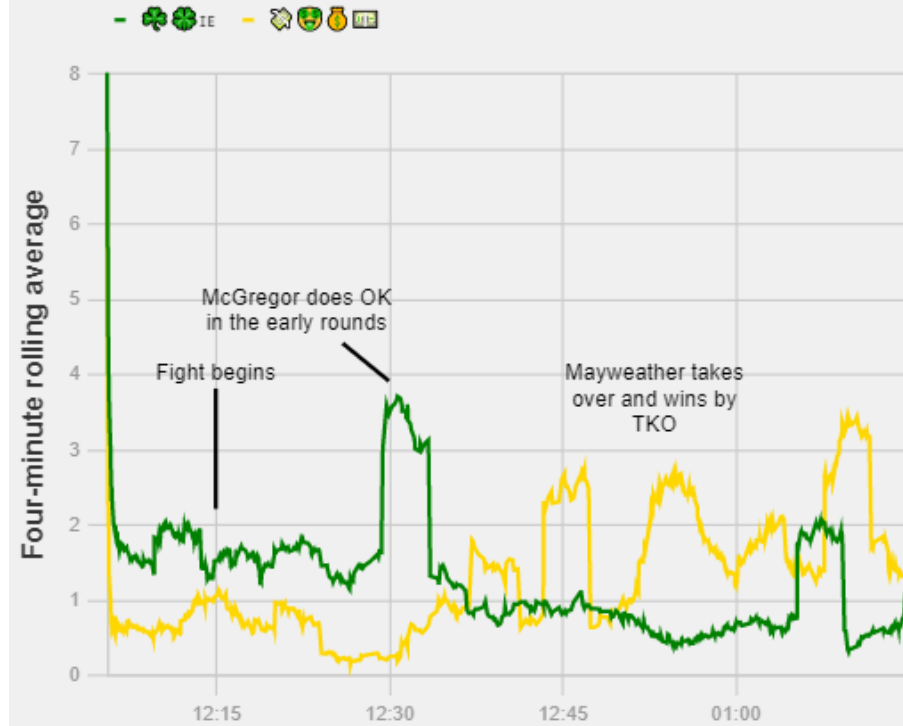
#raise NotImplementedError()

```

Out[13]:

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



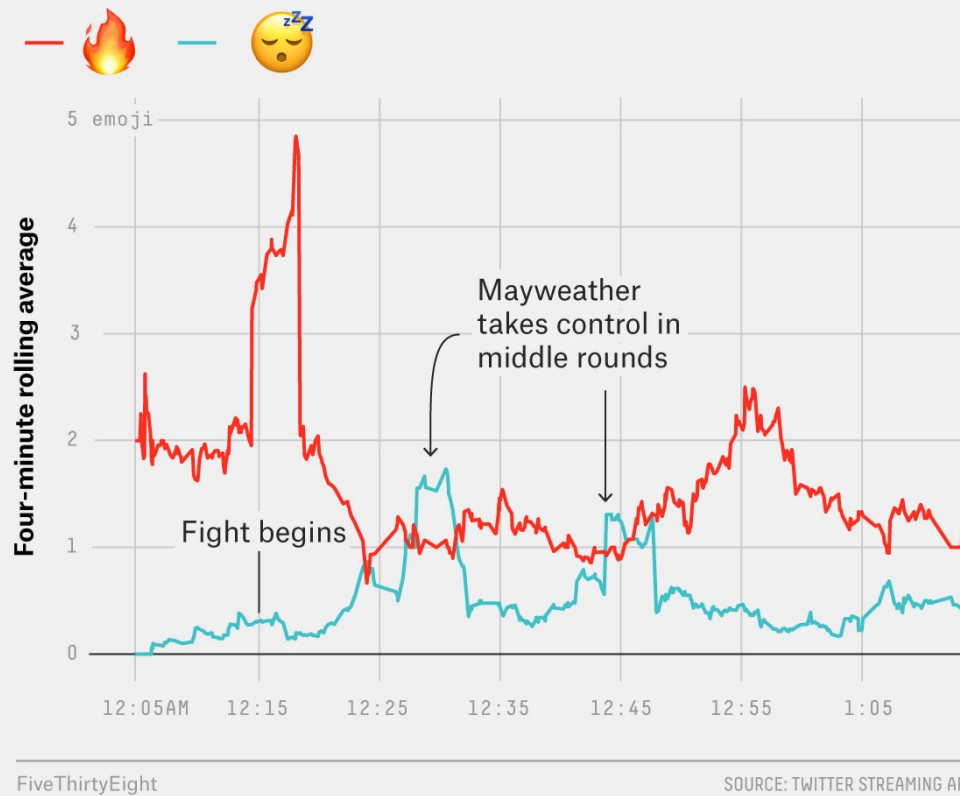
To the surprise of many (of the neutral and pro-Mayweather viewers, anyway) McGregor won the first round. The next couple were washes, and a quarter of the way into the [scheduled 12 rounds](https://www.nytimes.com/2017/08/26/sports/mayweather-mcgregor.html) ... the Irish underdog may have been winning! 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



** 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 [14]: *# your solution goes here, use the example above for the sampling and annotation*

```
teams_23 = tweets.copy()

teams_23['fire'] = tweets['🔥']
teams_23['sleep'] = tweets['😴']
teams_23 = teams_23.resample('1s').sum()
teams_23 = teams_23[(teams_23['🔥']>0) | (teams_23['😴']>0)]

# next we're going to creat a rolling average
# first for the money team
fire = teams_23['fire'].rolling('4Min').mean().reset_index()
fire['team'] = '🔥'
fire = fire.rename(columns={'fire':'tweet_count'})

# next for the irish team
sleep = teams_23['sleep'].rolling('4Min').mean().reset_index()
sleep['team'] = '😴'
sleep = sleep.rename(columns={'sleep':'tweet_count'})

# now we'll combine our datasets
ndf_23 = pd.concat([fire,sleep])

annotations_23 = [['2017-08-27 00:15:00',1.25, 'Fight begins'],
                  ['2017-08-27 00:40:00',3, 'Mayweather takes control in middle rounds']]
a_df_23 = pd.DataFrame(annotations_23, columns=['date','count','note'])

#raise NotImplementedError()

vis_23 = alt.Chart(ndf_23).mark_line().encode(
    y=alt.Y('tweet_count:Q', title = 'Four-minute rolling average'),
    x=alt.X('datetime:T', axis=alt.Axis(format='%I:%M', tickCount = 4)),
    color = alt.Color('team', legend = alt.Legend(orient = 'top', symbolType = 'stroke'), title = None,
        scale = alt.Scale(range = ['red', '#87CEEB'])))
).properties(
    title={
        "text": ["Much hype, some boredom"],
        "subtitle": ["Four-minute rolling average of the number of uses of selected emoji in",
            "sampled tweets during the Mayweather-McGregor fight"]
    }
)
```

```

ann_23 = alt.Chart(a_df_23).mark_text().encode(
    y=alt.Y('count:Q'),
    x=alt.X('date:T', axis=alt.Axis(format='%I:%M', tickCount = 4, title = None)),
    text = 'note:N'
)

lines_23 = [
    ['2017-08-27 00:15:00', 0.5, 'A'],
    ['2017-08-27 00:15:00', 1.1, 'A'],
    ['2017-08-27 00:30:00', 1.8, 'B'],
    ['2017-08-27 00:30:00', 2.8, 'B'],
    ['2017-08-27 00:45:00', 1.5, 'C'],
    ['2017-08-27 00:45:00', 2.8, 'C']]

l_df_23 = pd.DataFrame(lines_23, columns=['x', 'y', 'group'])

l_23 = alt.Chart(l_df_23).mark_line().encode(
    y=alt.Y('y:Q'),
    x=alt.X('x:T', axis=alt.Axis(format='%I:%M', tickCount = 4, title = None)),
    detail = 'group',
    color = alt.value('black')
)

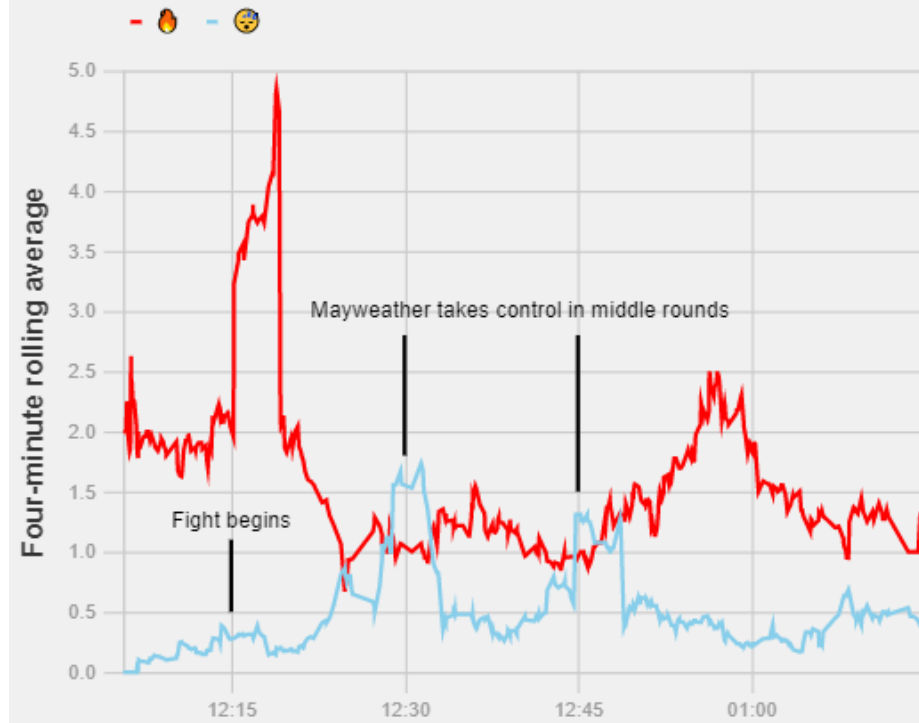
vis_23 + ann_23 + l_23

```

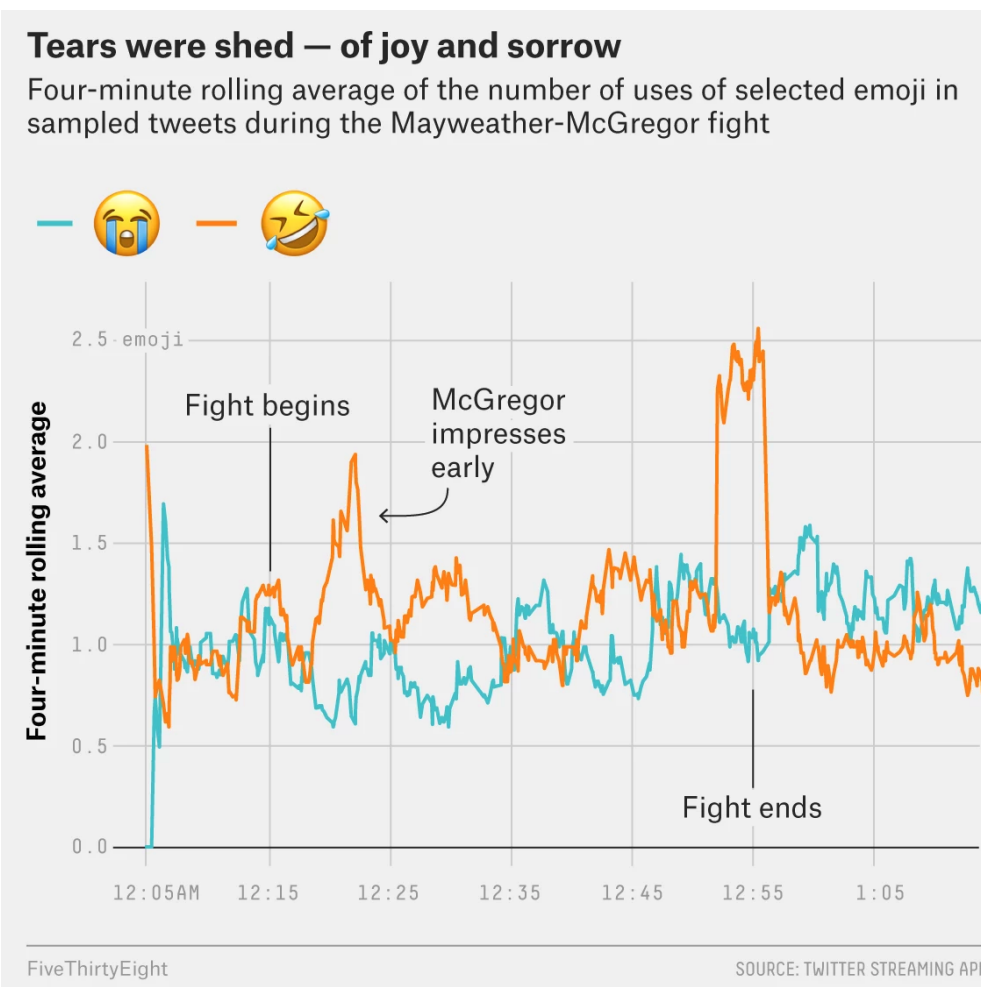
Out[14]:

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/) (<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) (<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-fight-stoppage-let-the-man-put-me-down/) (<https://www.cbssports.com/boxing/news/conor-mcgregor-frustrated-with-refs-fight-stoppage-let-the-man-put-me-down/>). 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 🤔🤔🤔🤔.



** 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 [15]: # your solution goes here, use the example above for the sampling and annotation
teams_24 = tweets.copy()

teams_24['down'] = tweets['🥊']
teams_24['side'] = tweets['🥋']
teams_24 = teams_24.resample('1s').sum()
teams_24 = teams_24[(teams_24['🥊']>0) | (teams_24['🥋']>0)]

# next we're going to creat a rolling average
# first for the money team
down = teams_24['down'].rolling('4Min').mean().reset_index()
down['team'] = '🥊'
down = down.rename(columns={'down':'tweet_count'})

# next for the irish team
side = teams_24['side'].rolling('4Min').mean().reset_index()
side['team'] = '🥋'
side = side.rename(columns={'side':'tweet_count'})

# now we'll combine our datasets
ndf_24 = pd.concat([down,side])

annotations_24 = [['2017-08-27 00:15:00',2.1, 'Fight begins'],
                  ['2017-08-27 00:35:00',2, 'McGregor \nimpresses \nearly'],
                  ['2017-08-27 00:56:00',0.1, 'Fight ends']]
a_df_24 = pd.DataFrame(annotations_24, columns=['date','count','note'])

#raise NotImplementedError()

vis_24 = alt.Chart(ndf_24).mark_line().encode(
    y=alt.Y('tweet_count:Q', title = 'Four-minute rolling average'),
    x=alt.X('datetime:T', axis=alt.Axis(format='%I:%M', tickCount = 4, title = None)),
    color = alt.Color('team', legend = alt.Legend(orient = 'top', symbolType = 'stroke'), title = None,
        scale = alt.Scale(range = ['#00CED1', '#FF8C00']))
).properties(
    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 during the Mayweather-McGregor fight"]
    }
)

```

```

ann_24 = alt.Chart(a_df_24).mark_text(lineBreak = '\n').encode(
    y=alt.Y('count:Q'),
    x=alt.X('date:T', axis=alt.Axis(format='%I:%M', tickCount = 4, title = None)),
    text = 'note:N'
)

lines_24 = [
    ['2017-08-27 00:15:00', 1.4, 'A'],
    ['2017-08-27 00:15:00', 2.0, 'A'],
    ['2017-08-27 00:24:00', 1.8, 'B'],
    ['2017-08-27 00:29:00', 2.0, 'B'],
    ['2017-08-27 00:56:00', 0.2, 'C'],
    ['2017-08-27 00:56:00', 0.9, 'C']]

l_df_24 = pd.DataFrame(lines_24, columns=['x', 'y', 'group'])

l_24 = alt.Chart(l_df_24).mark_line().encode(
    y=alt.Y('y:Q'),
    x=alt.X('x:T', axis=alt.Axis(format='%I:%M', tickCount = 4, title = None)),
    detail = 'group',
    color = alt.value('black')
)

vis_24 + ann_24 + l_24

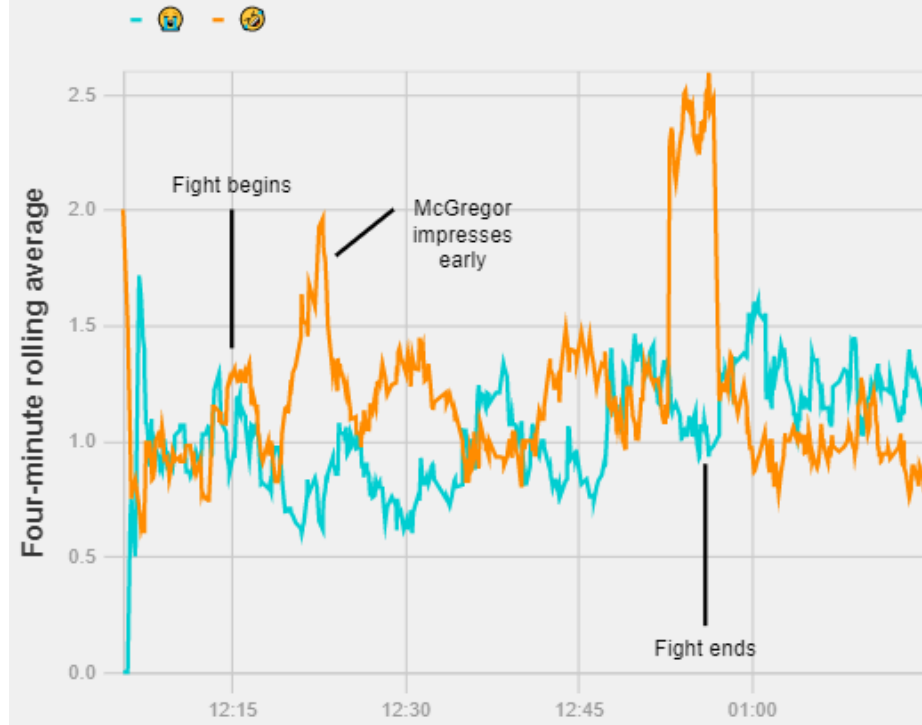
#raise NotImplementedError()

```

Out[15]:

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.

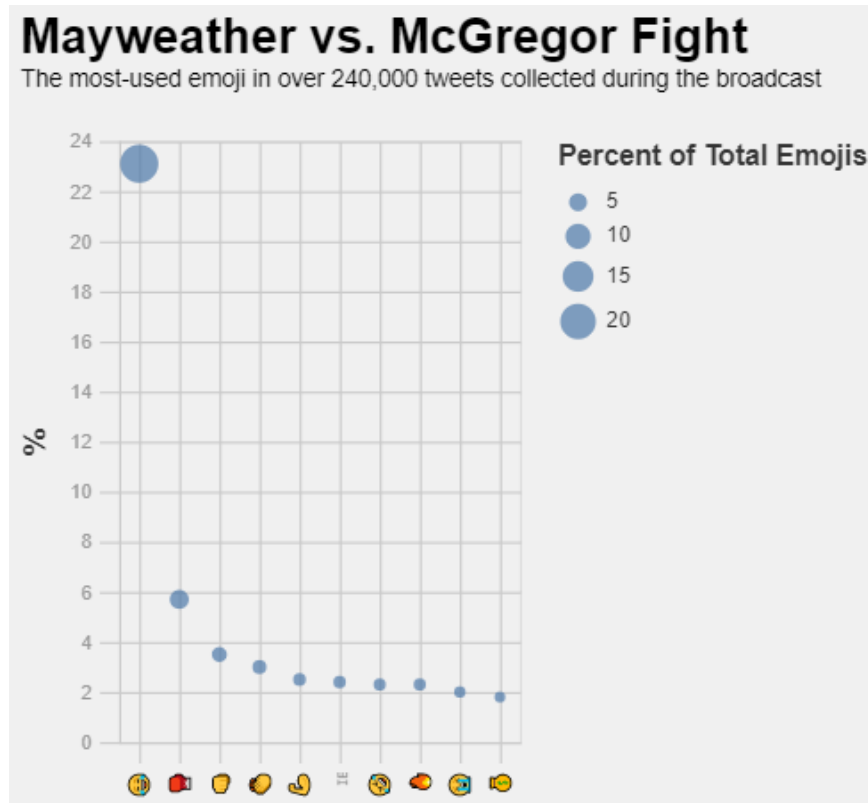
2.5 Make your own (part 1-alternative)

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 [16]: viz_25 = alt.Chart(percentages_df).mark_point().encode(
x = alt.X('EMOJI', sort = sorted_emoji, title = None),
y = alt.Y('PERCENT', title = '%'),
size = alt.Size('PERCENT', legend = alt.Legend(symbolType = 'circle'), title = 'Percent of Total Emojis')
).properties(
    title = {"text": ["Mayweather vs. McGregor Fight"],
            "subtitle": ["The most-used emoji in over 240,000 tweets collected during the broadcast"]})

viz_25
#raise NotImplementedError()
```

Out[16]:

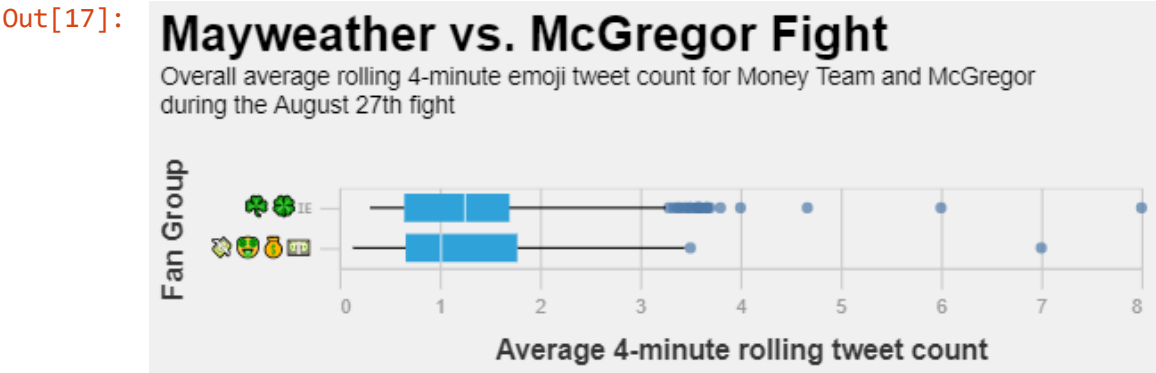


This is a recreated visual based on the bar graph featured in the article and re-created in 2.1. The figure has the top 10 emojis encoded on the x-axis, the percentage of the total usage compared to the other emojis encoded on the y-axis, and this total relative usage percentage is also encoded as the size of the individual bubble corresponding to each of the emojis. However, I believe as though this figure is not as effective as the figure used in the article and re-created earlier. While the figures do have the same level of expressiveness (as the percentage of the emoji's relative usage can be discerned through both figures), the original figure is far more effective. As discussed throughout the course and in terms of perception and cognition, Stevens' Law makes clear that viewers are very adept at actually determining a value based on length, and tend to underestimate a value when area or volume is used. The original figure utilizes bar graph length as its value encoding, where as area is used here, rendering a less effective figure. Additionally, and as noted earlier, as well, this is an example of Munzner's Rule of thumb that if a list is good enough, a 2D image is not justified. I believe that the best way to encode this data would be a simple table, as opposed to a bar graph, scatter plot, etc. as the data can very easily be deciphered through this presentation.

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 [17]: alt.Chart(ndf).mark_boxplot().encode(
    y= alt.Y('team:N', title = 'Fan Group'),
    x= alt.X('tweet_count:Q', title = 'Average 4-minute rolling tweet count')
).properties(
    title = {"text": ["Mayweather vs. McGregor Fight"],
            "subtitle": ["Overall average rolling 4-minute emoji tweet count for Money Team and McGregor",
                        "during the August 27th fight"]}]
)
```



While other figures in the article mainly focused around timeseries data and tracking rolling averages of emoji use throughout the Mayweather and McGregor fight, this new figure generates a box and whisker plot that focuses on emoji usage tied to the Money Team and the McGregor camp. A total average 4-minute rolling tweet usage quantitative variable is encoded on the x-axis and the two emoji groups are nominal variables encoded in the y position. As can be seen, there are multiple cases throughout the fight where the Money Team

and McGregor fans had emoji usage that were considered "outliers" to the box and whisker plot. I believe this is an effective plot in showing that, throughout the night, McGregor looked to be the more popular of the two boxers and often had outlier rolling averages, which likely are aligned with the period of the night where he performed better than expected. In terms of perception and cognition, a box and whisker plot has been a discussion point in relation to the data-ink ratio. While there perhaps may be some unnecessary ink added to the plot, viewers have been shown to be more adept at interpreting these plots as opposed to others which may have unneeded components removed. Therefore, while there may be other plot alternatives that would have equal expressiveness, I believe this is an effective plot in displaying overall emoji usage.