

# Information Visualization I

## School of Information, University of Michigan

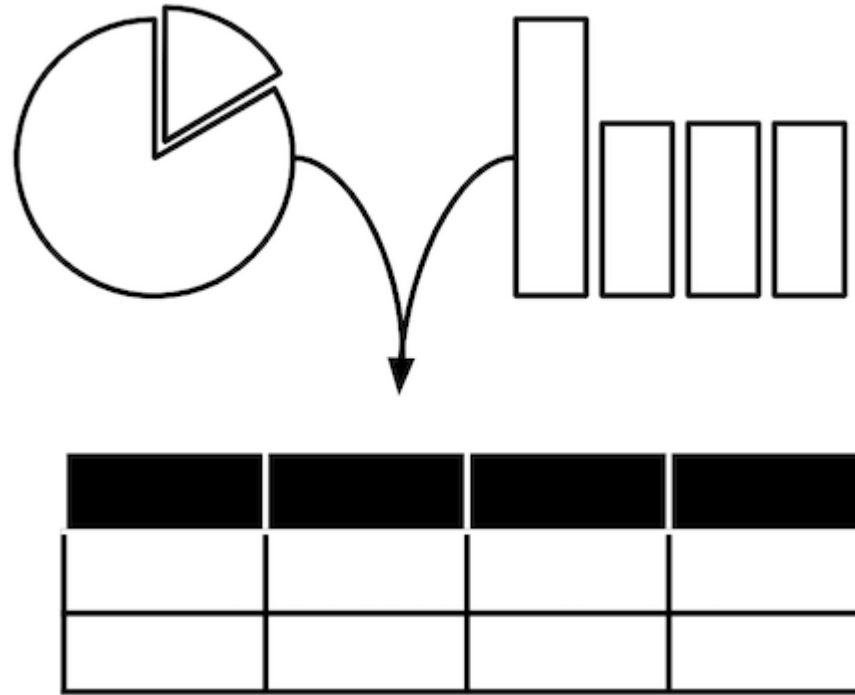
### Week 2:

- Expressiveness and Effectiveness
- Grammar of Graphics

### Assignment Overview

#### Our objectives for this week:

- Review, reflect, and apply the concepts of encoding. Given a visualization recreate the data that was encoded.
- Review, reflect, and apply the concepts of Expressiveness and Effectiveness. Given a visualization, evaluate alternatives with the same expressiveness.



Two visualizations, same expressiveness

- Review and evaluate an implementation of Grammar of Graphics 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: Next Bechdel Test (30 points)
- Altair programming exercise (70 points)
- Bonus (5 points)

### Resources:

- This article by [FiveThirtyEight](https://fivethirtyeight.com) (<https://fivethirtyeight.com>) available [online](https://projects.fivethirtyeight.com/next-bechdel/) (<https://projects.fivethirtyeight.com/next-bechdel/>) (Hickey, Koeze, Dottle, Wezerek 2017)
- Datasets from FiveThirtyEight, we have downloaded a subset of these datasets available in the folder for your use into [./assets](#) ([./assets](#))
  - The original dataset can be found on [FiveThirtyEight Next Bechdel Dataset](https://github.com/fivethirtyeight/data/tree/master/next-bechdel) (<https://github.com/fivethirtyeight/data/tree/master/next-bechdel>).

## Important notes:

- 1) Grading for this assignment is largely done by manual inspection. The autograder checks the basic details but is imperfect. Sometimes it will pass the test but not look right, sometimes it will look right and fail the test. That's fine! You should guide your answer by the look of our examples. It doesn't need to be pixel perfect (e.g., you may not always know what our example is scaled by), but it should be pretty close.
- 2) 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.
- 3) Pay attention to the return types of your functions.

## Part 1. Expressiveness and Effectiveness (30 points)

Read the following article [Creating the next Bechdel Test \(https://projects.fivethirtyeight.com/next-bechdel/\)](https://projects.fivethirtyeight.com/next-bechdel/) and answer the following questions:

### 1.1 Recreate the table (by hand or excel) needed to create the following visualization (7 points)

You *should* consider the interactive parts of the visualization in your answer. Take a picture or screenshot of your table and add it to the answer below

## THE UPHOLD TEST

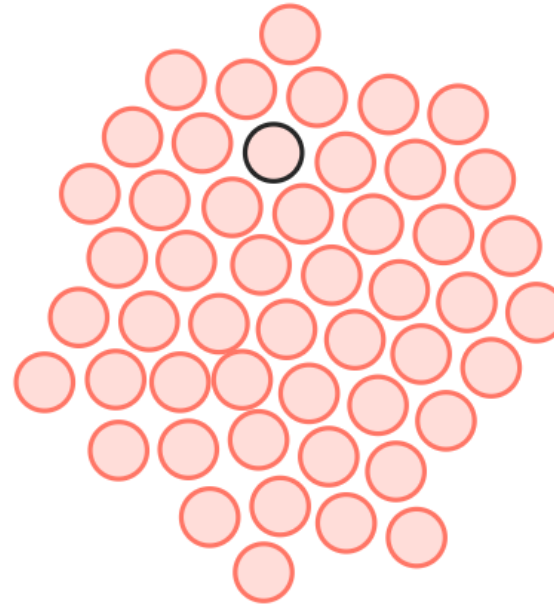
---

Rory Uphold: writer and actress on “This Is Why We’re Single”



### A movie passes if:

- The on-set crew is 50 percent women



0 passed

All 50 failed

---

➤ SHOW TEST NOTES

### ○ THE CONJURING 2 BY THE NUMBERS

Using names to estimate gender, “The Conjuring 2” was one of the worst offenders, with men accounting for about 90 percent of the crew.

An easy way to upload images is to jump into the [./assets](#) ([./assets](#)) directory (or use the Coursera notebook explorer and navigate to it) and then use the upload button to save your image:



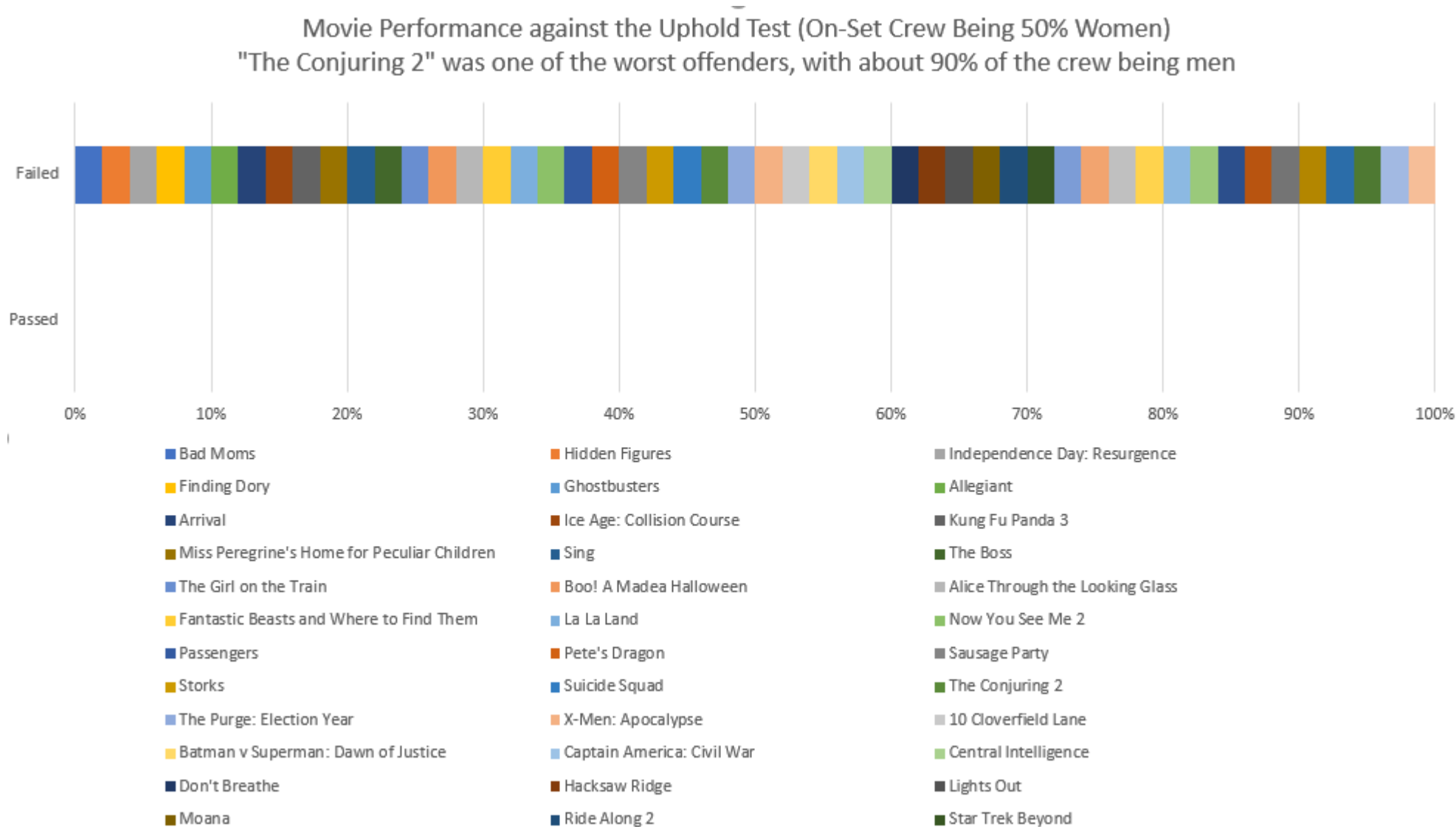
Once you have the image, you can link to it using the markdown command: `![answer1.2](assets/my_image_1.2.png)`

movie	failed_uphold_test	on_set_crew_men_pct
Bad Moms	1	?
Hidden Figures	1	?
Independence Day: Resurgence	1	?
Finding Dory	1	?
Ghostbusters	1	?
Allegiant	1	?
Arrival	1	?
Ice Age: Collision Course	1	?
Kung Fu Panda 3	1	?
Miss Peregrine's Home for Peculiar Children	1	?
Sing	1	?
The Boss	1	?
The Girl on the Train	1	?
Boo! A Madea Halloween	1	?
Alice Through the Looking Glass	1	?
Fantastic Beasts and Where to Find Them	1	?
La La Land	1	?
Now You See Me 2	1	?
Passengers	1	?
Pete's Dragon	1	?
Sausage Party	1	?
Storks	1	?
Suicide Squad	1	?
The Conjuring 2	1	~90%
The Purge: Election Year	1	?
X-Men: Apocalypse	1	?
10 Cloverfield Lane	1	?
Batman v Superman: Dawn of Justice	1	?
Captain America: Civil War	1	?
Central Intelligence	1	?
Don't Breathe	1	?
Hacksaw Ridge	1	?

The table (seen above) will have three columns needed for the visualization: the movie name, whether the movie passed the Uphold Test, and the percentage of men that were crew members (due to the Conjuring 2 note). However, this third value is only expressed for this one movie, leading to an inability to derive the value for the other 49 movies.

1.2 Sketch an alternative visualization with the same expressiveness (7 points)













By hand is fine, but you can also use a tool. This is a sketch, the data need not be perfectly accurate or to scale. Again, upload a picture or screenshot below. Make sure there is enough annotation so it's clear why your picture has the same expressiveness.



This alternative visualization has the same expressiveness as the prior as it shows all 50 movies that went through the Uphold Test, whether they passed or failed, and gave detail

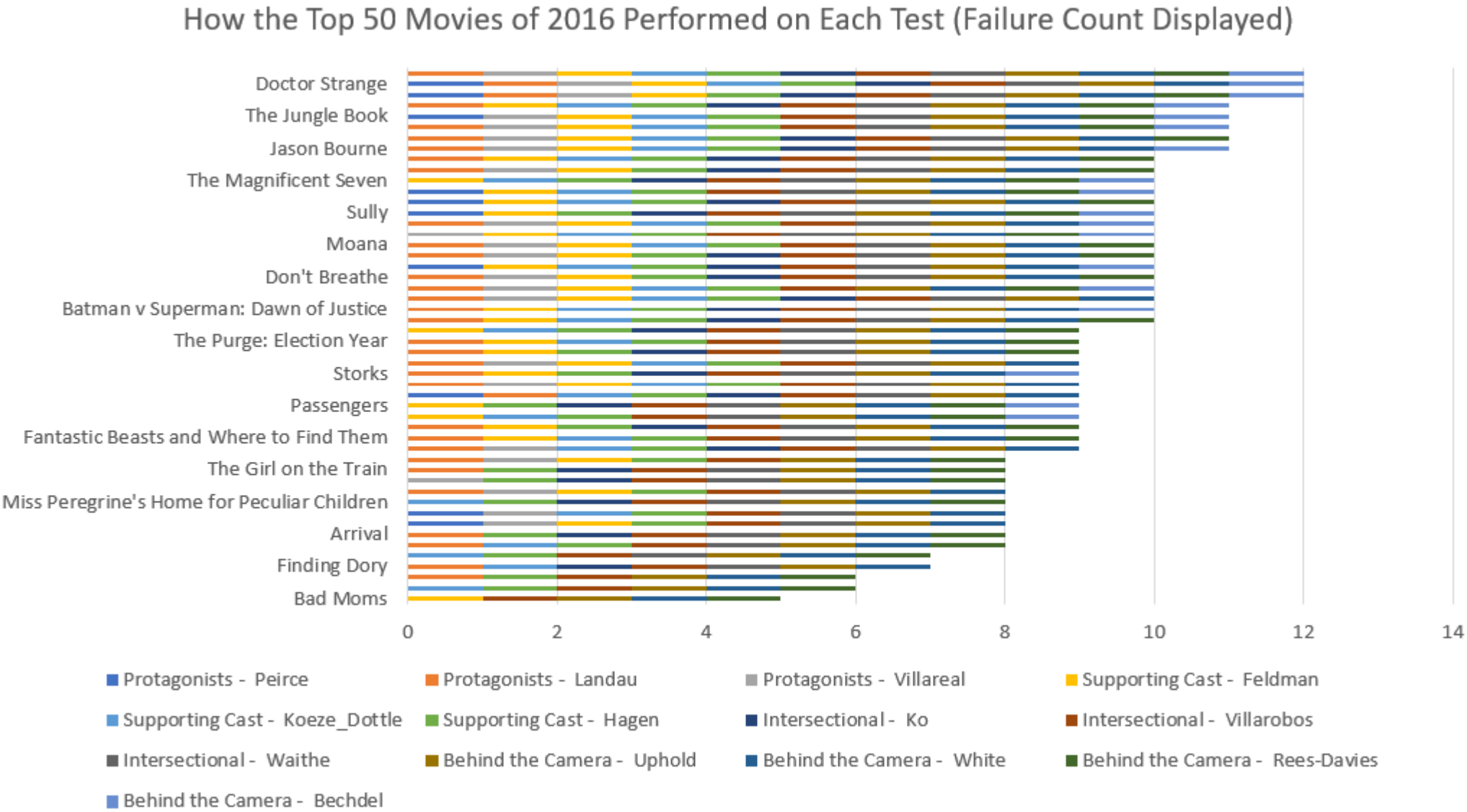
around what the Uphold Test entails. The note about "The Conjuring 2" being a poor performer was again pulled forward.

1.3 Sketch an alternative visualization with the same expressiveness of the following visualization (10 points)

MOVIES	TEST											
	BEHIND THE CAMERA				INTERSECTIONAL			PROTAGONISTS			SUPPORTING CAST	
												
Bad Moms	✓				✓	✓		✓	✓	✓	✓	✓
Hidden Figures	✓				✓	✓		✓	✓	✓		✓
Independence Day: Resurgence	✓				✓	✓		✓	✓		✓	✓
Finding Dory	✓		✓					✓	✓		✓	✓
Ghostbusters	✓					✓		✓	✓	✓		✓
Allegiant	✓					✓		✓	✓			✓
Arrival	✓							✓	✓		✓	✓
Ice Age: Collision Course	✓		✓			✓				✓		✓



Same deal as last question: by hand or with a tool is fine. The data need not be perfectly accurate or to scale. Make sure there is enough annotation so it's clear why your picture has the same expressiveness. Again, upload a picture or screenshot below.



1.4 Reflect on which visualization you think is more *effective* and why? (6 points)

You are comparing the original figure in 1.3 and the one you created.

The original figure and the new generated figure would have the same level of expressiveness in that movie performance across all of the various tests is displayed. The viewer would be able to determine which tests a movie was able to pass, and gives detail in regards to the higher-level group the test belonged to (Bechdel being a Behind the Camera test, for example).

However, I believe the original image is a far more effective figure. The way in which the data is displayed is far easier to digest, as you can much more quickly assess which tests a movie did or did not pass. For the other figure, while this is still possible to discern, referencing the color tied to each test in the legend is time consuming. The overall presentation of the original figure is also much cleaner, and a sorted visualization also is very useful in quickly deciphering which movies are high or low performers.

## Part 2. Altair programming exercise (70 points)

We have provided some code to create visualizations based on the following datasets:

1. [all\\_tests \(assets/nextBechdel\\_allTests.csv\)](#) Is a collection of different Bechdel test results for the 50 top-grossing films [at the domestic box office in 2016 \(https://www.the-numbers.com/box-office-records/domestic/all-movies/cumulative/released-in-2016\)](https://www.the-numbers.com/box-office-records/domestic/all-movies/cumulative/released-in-2016)
2. [cast\\_gender \(assets/nextBechdel\\_castGender.csv\)](#) Is the gender for all the cast member of each movie in the Bechdel rankings
3. [top\\_2016 \(assets/top\\_2016.csv\)](#) Is the date, box office and theater count for each top 2016 movie.

Complete each assignment function and run each cell to generate the final visualizations

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

```
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]: # read all the tables
all_tests_df = pd.read_csv('assets/nextBechdel_allTests.csv')
cast_gender = pd.read_csv('assets/nextBechdel_castGender.csv')
top_2016 = pd.read_csv('assets/top_2016.csv')
```

```
In [5]: # set up the tables for use
actors_movies = top_2016.set_index('Movie').join(cast_gender.set_index('MOVIE')).join(all_tests_df.set_index('movie')).reset_index().dropna()
movies_order = top_2016.sort_values(by=['Rank'])['Movie'].tolist()
```

## 2.1 Variables encoded (5 points)

Warmup: complete the `variables_encoded` function. Return the number of variables encoded in the following example

```
In [6]: base = alt.Chart(actors_movies).transform_filter(
    (alt.datum.TYPE != 'Unknown') & (alt.datum.GENDER != 'Unknown') & (alt.datum.GENDER != 'null')
)
```

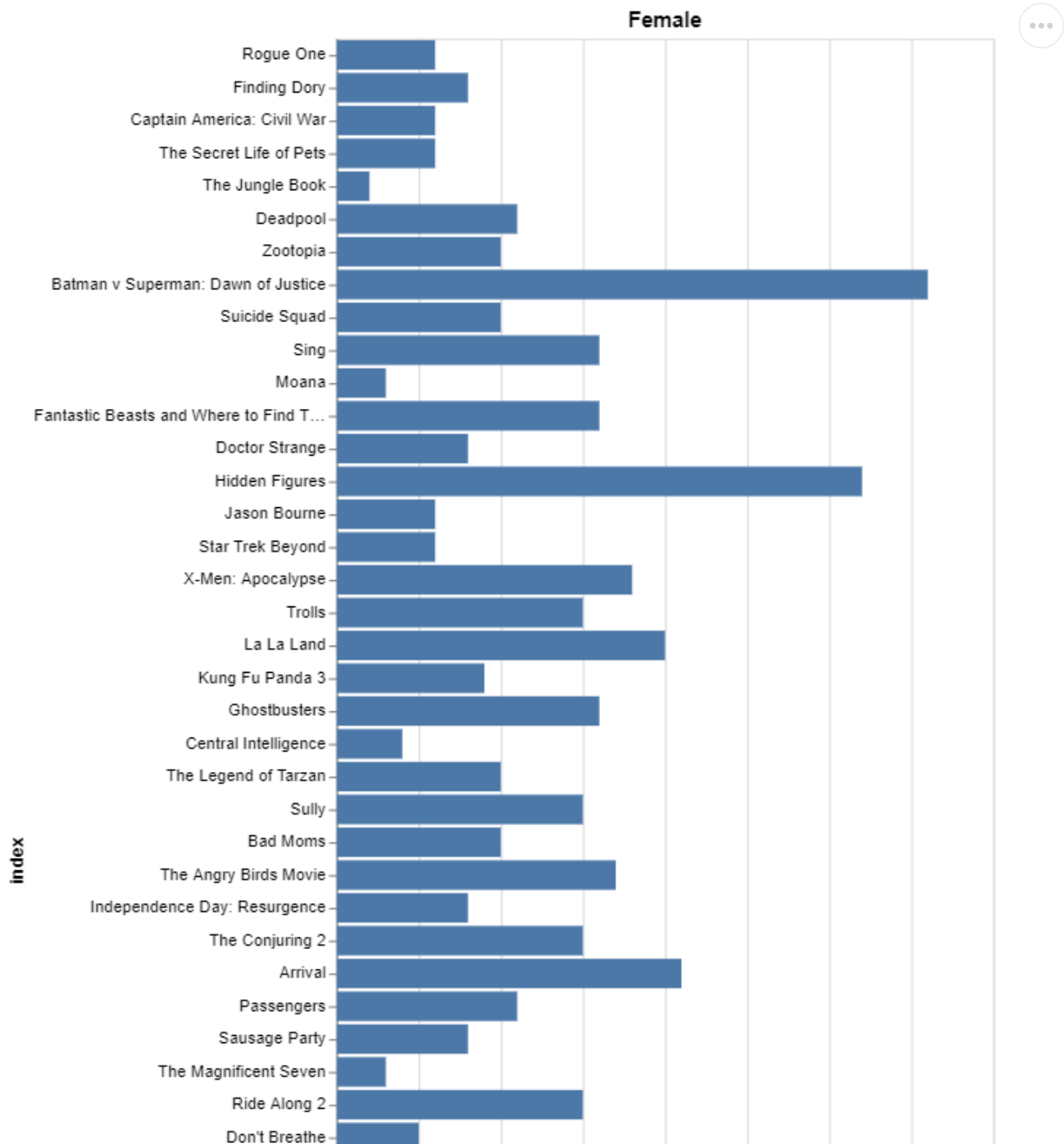
```
In [7]: ##Note that the numbers below are the running tally of encoded variables to show work. This is done throughout the assignment.
encoding = base.transform_filter(
    alt.datum.GENDER == 'Female'
).encode(
    y= alt.Y(
        'index:N', #1, 2 (nominal designation)
        sort= movies_order#3
    ),
    x=alt.X('count(index):Q', #4, 5 (quantitative designation)
        title='cast count'),
)

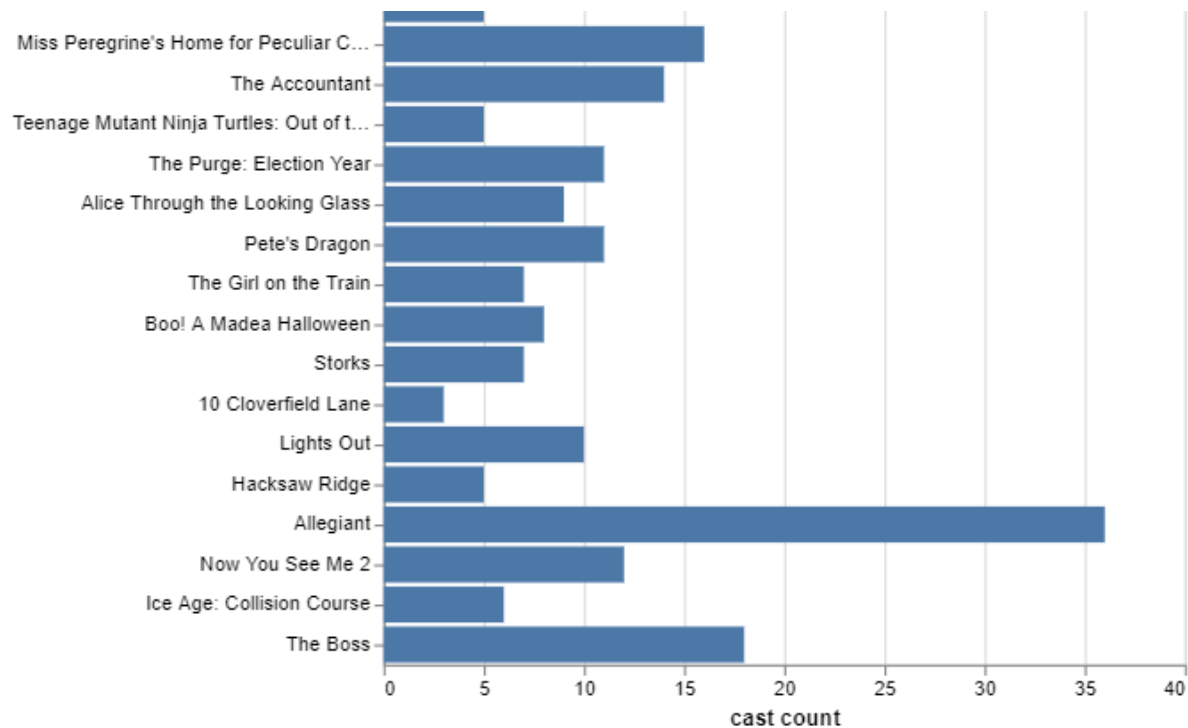
def m_bar():
    return encoding.mark_bar().properties(title='Female')

bar = m_bar()
```

```
In [8]: bar
```

Out[8]:



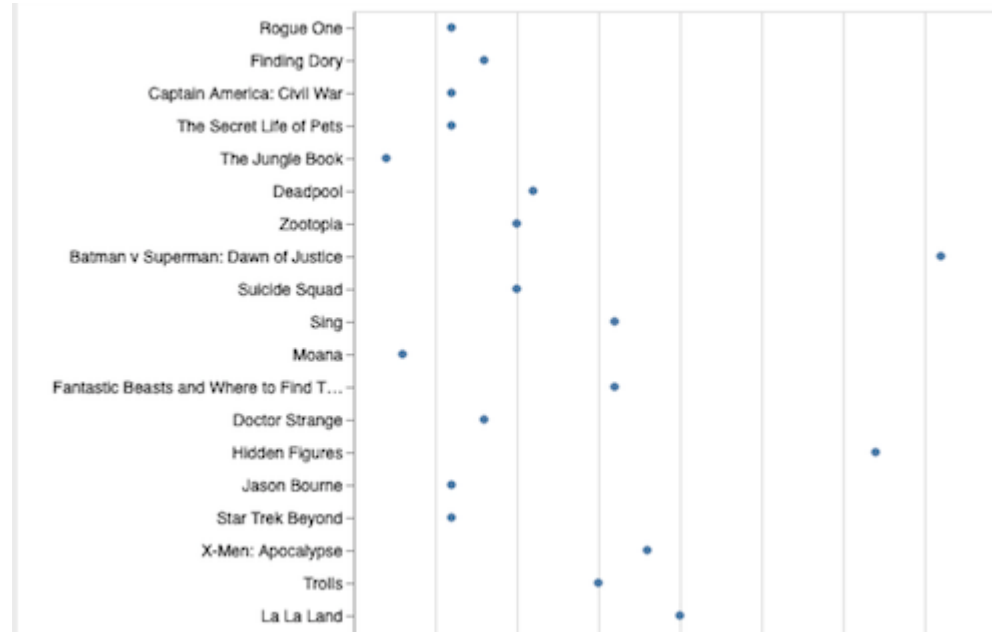


```
In [9]: def variables_encoded():
        """
        This is a helper function for automated grading, just return the number (e.g., 'return 50000;'). You don't
        need to do anything fancy here.
        """
        return 5
        #raise NotImplementedError()
```

```
In [10]: #hidden tests are within this cell
```

## 2.2 Alternative encoding (5 points)

Complete the `m_circle` function. Change the encoding used in the previous example from a bar to a circle. Your visualization should look like

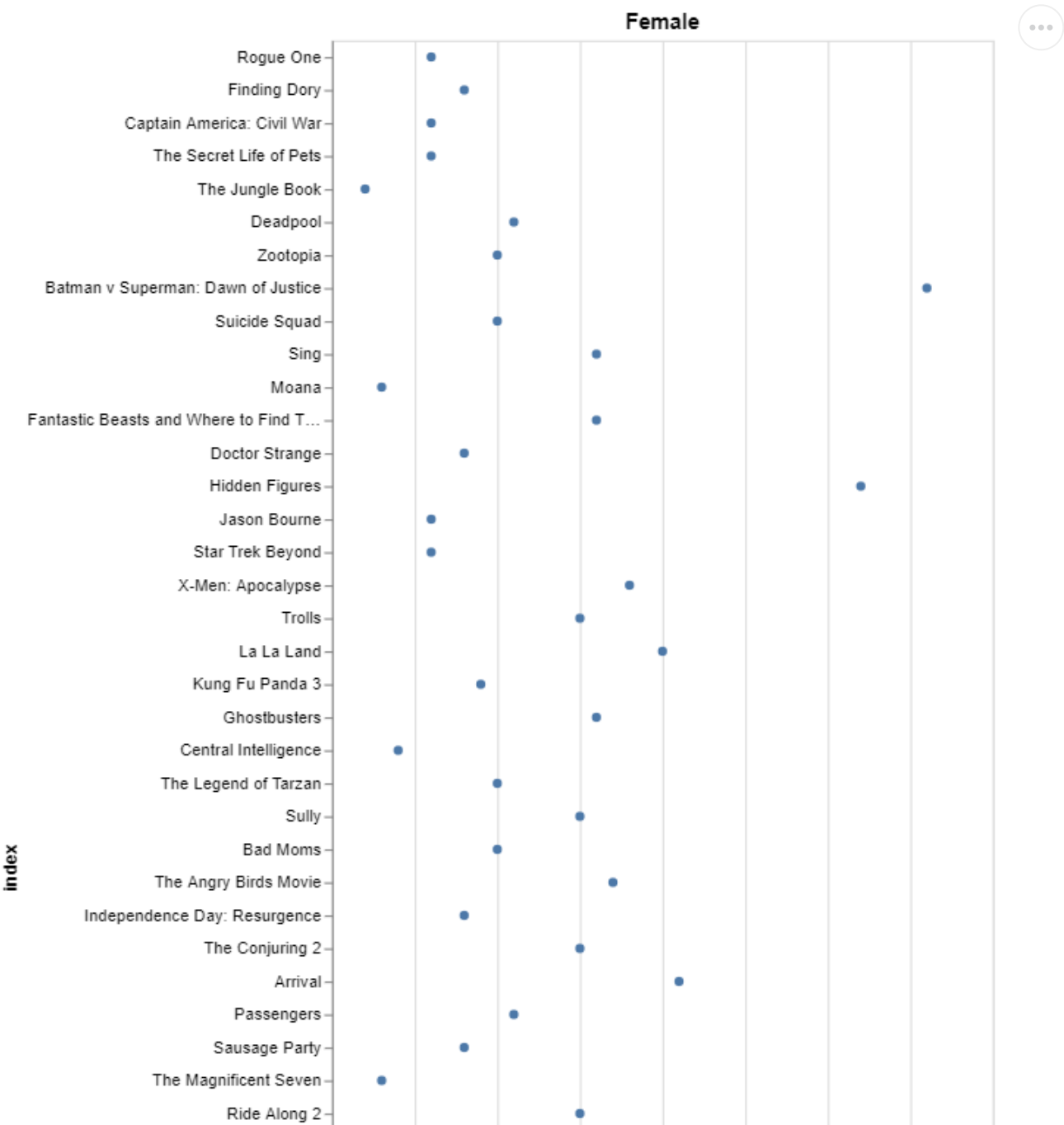


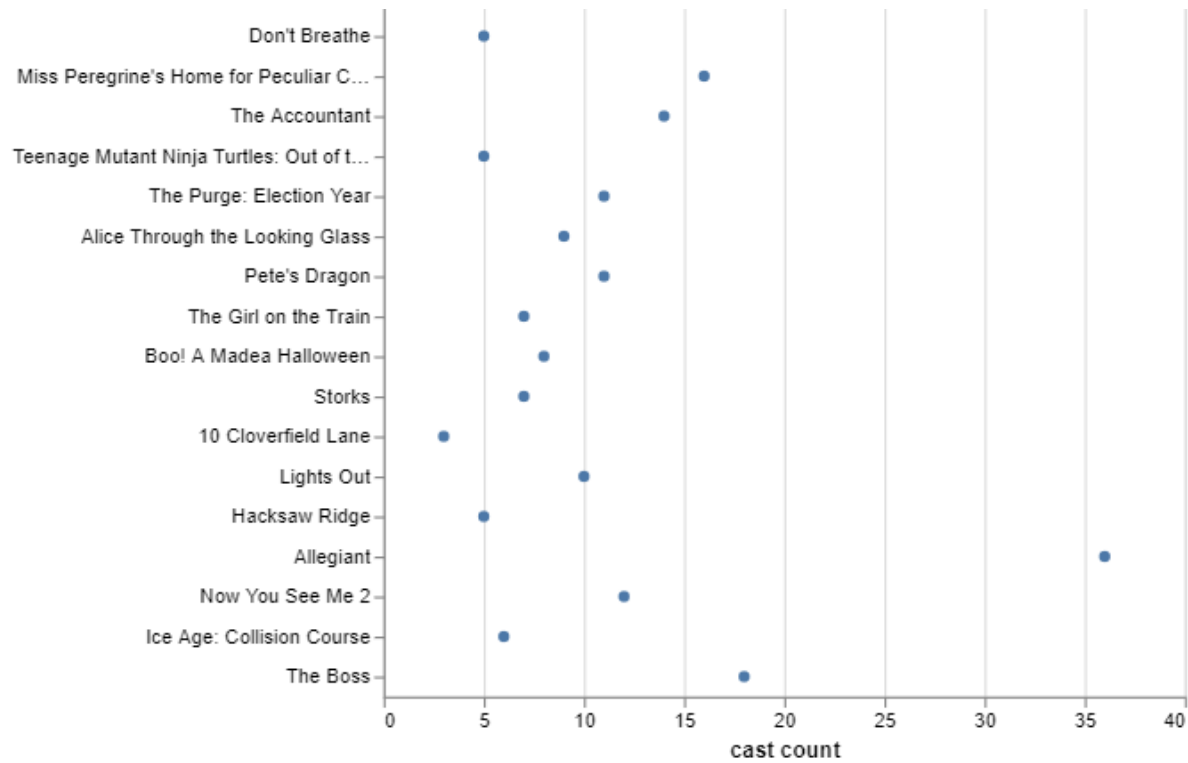
click [here \(assets/alt\\_enc\\_1\\_resized\\_fullImage.png\)](#) to see the full-sized image.

```
In [11]: def m_circle():  
    """  
    return the call to altair function that uses the circle mark for the variables encoded in the previous example  
    """  
    return encoding.mark_circle(filled = True).properties(title='Female')  
    #raise NotImplementedError()
```

```
In [12]: circle = m_circle()  
circle
```

Out[12]:



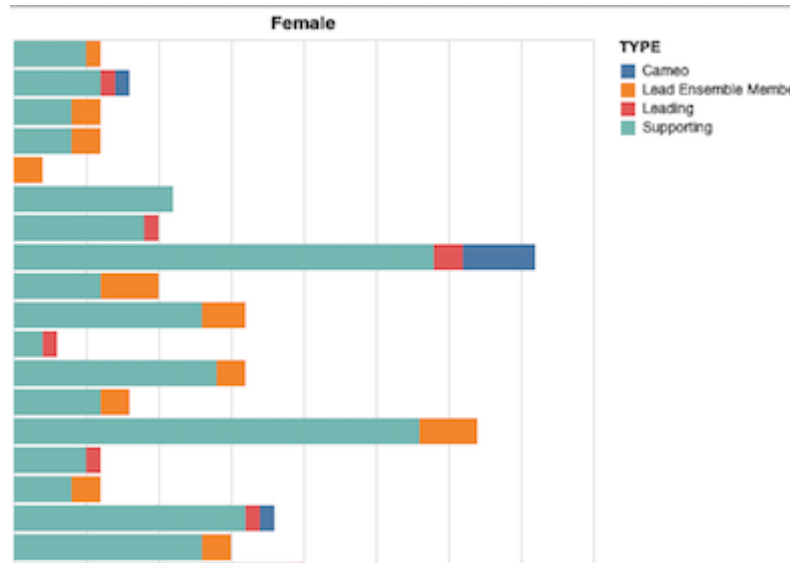


In [13]: *#hidden tests are within this cell*

## 2.3 Increase variables encoded (5 points)

Complete the `female_actors` function. Modify the first bar chart encoding the type of the actor with the color of the bar. Your visualization should look like the following (note that we don't have labels yet):





click [here \(assets/alt\\_enc\\_2\\_fullSize.png\)](#) to see the full-sized image.

- *Partial credit can be granted for each visualization (up to 2 points) if you provide a description of what the missing piece of the function is supposed to do without need for an Altair working version*

```
In [14]: def female_actors():
        """
        return the call to Altair function that uses the bar mark for the variables and the color for the TYPE
        """

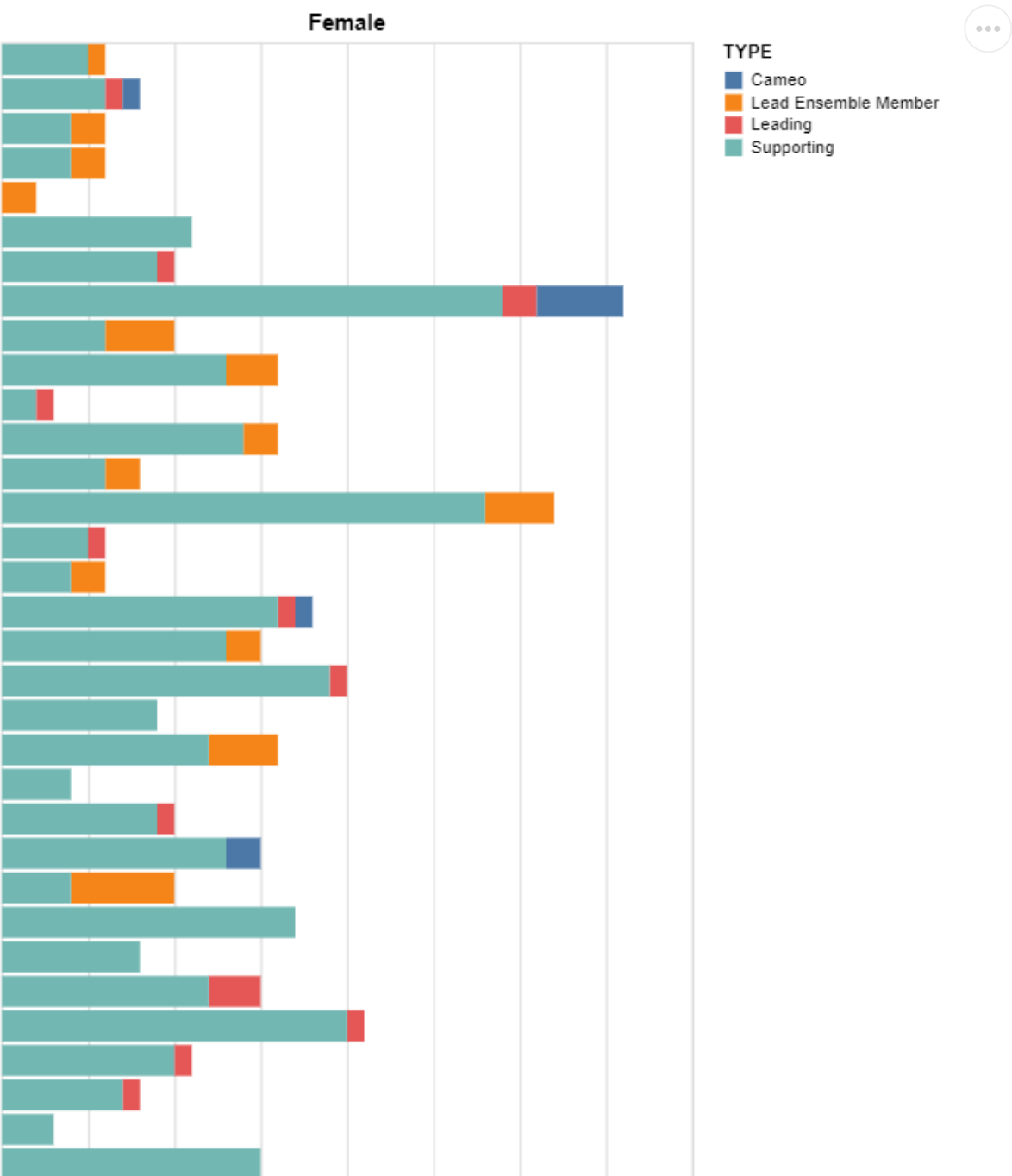
        female = base.transform_filter(
            alt.datum.GENDER == 'Female'
        ).encode(
            y= alt.Y(
                'index:N', #1, 2 (nominal designation)
                sort= movies_order, #3
                axis = None
            ),
            x=alt.X('count(index):Q', #4, 5 (quantitative designation)
                title='cast count'),
            color = alt.Color('TYPE') #6
        )

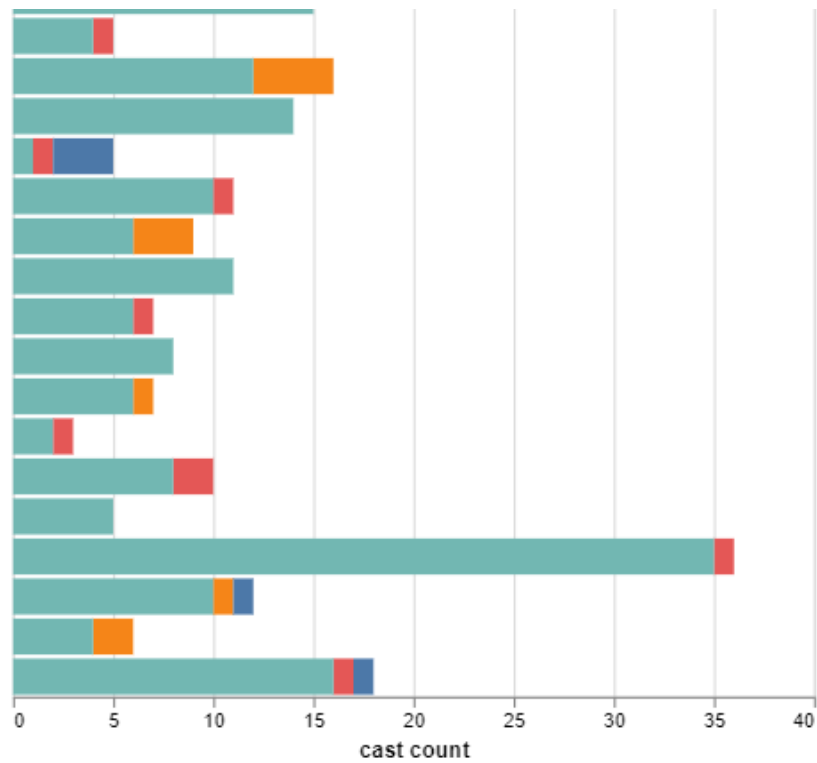
        #raise NotImplementedError()

        return female.mark_bar().properties(title='Female')
```

```
In [15]: female = female_actors()  
female
```

Out[15]:

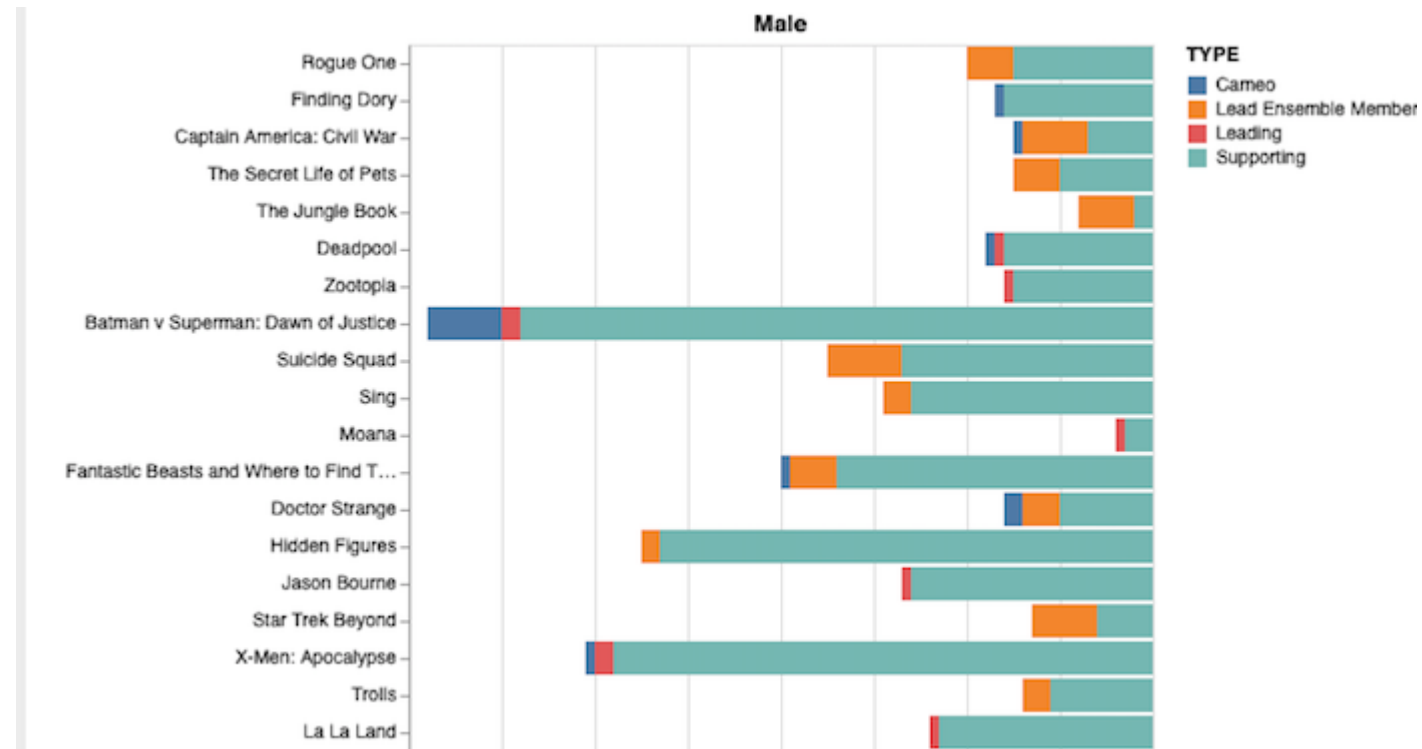




In [16]: *#hidden tests are within this cell*

## 2.4 Change filter transform (5 points)

Complete the `male_actors` function, modify the previous visualization so that the actors visualized have Male gender. Use the Altair transform function for this, not Pandas.



click [here \(assets/alt\\_enc\\_3\\_fullSize.png\)](#) to see the full-sized image.

- *Partial credit can be granted for each visualization (up to 2 points) if you provide a description of what the missing piece of the function is supposed to do without need for an altair working version*

```
In [17]: def male_actors():
        """
        return an altair-defined chart that uses the bar mark for the variables and the color for the TYPE
        this function filters the actors to be male
        """
        #add filter transform

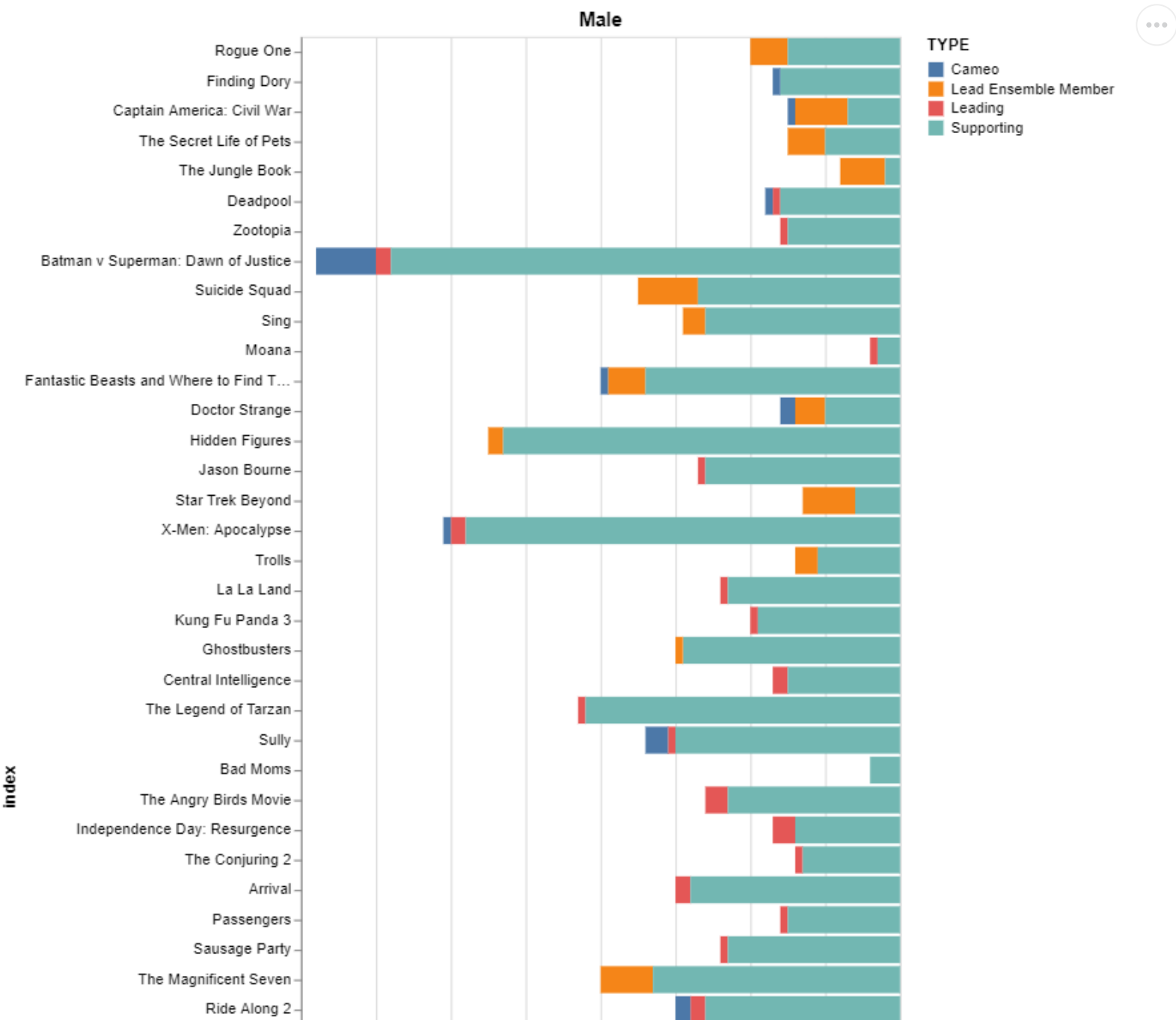
        # This is the starting point. Again, modify or replace this code to get the encoding we describe above
        male = base.transform_filter(
            alt.datum.GENDER == 'Male'
        ).encode(
            y= alt.Y(
                'index:N',#1, 2 (nominal designation)
                sort = movies_order#3
            ),
            x=alt.X('count(index):Q',#4, 5 (quantitative designation)
                title = 'cast count',
                sort = 'descending'),#6
            color = alt.Color('TYPE') #7
        )

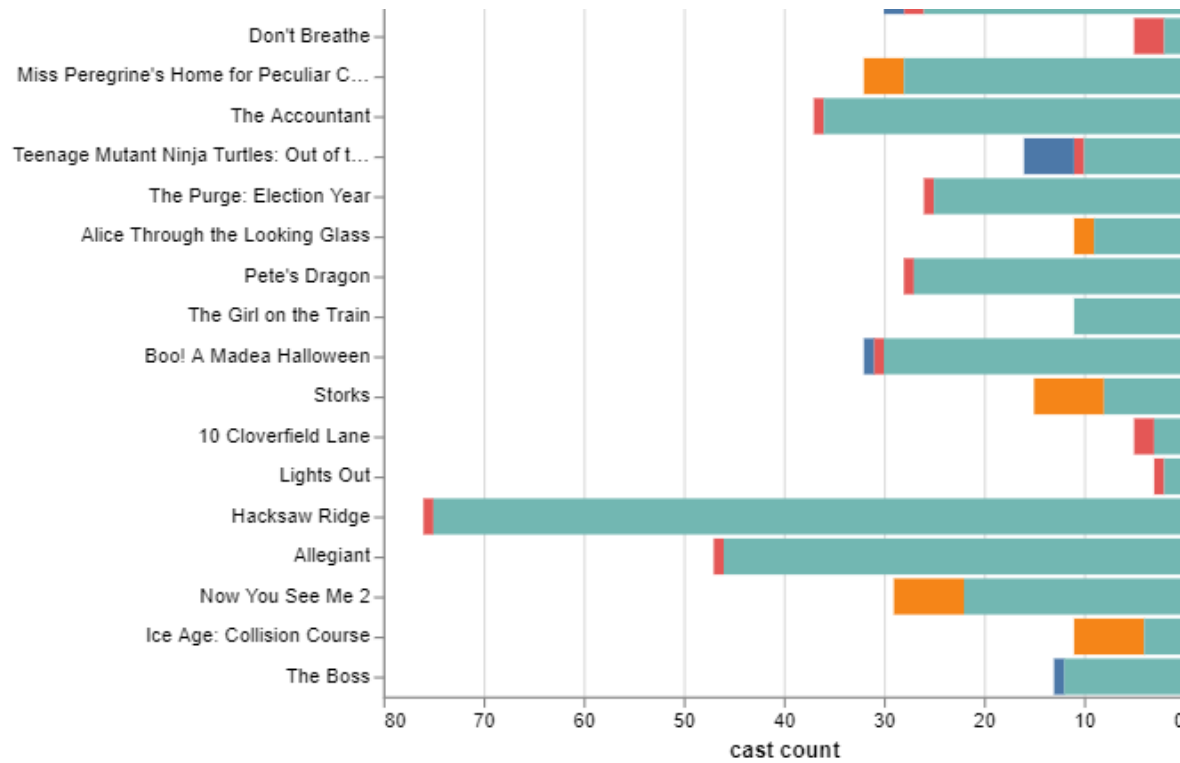
        #raise NotImplementedError()

        return male.mark_bar().properties(title='Male')
```

```
In [18]: male = male_actors()  
male
```

Out[18]:



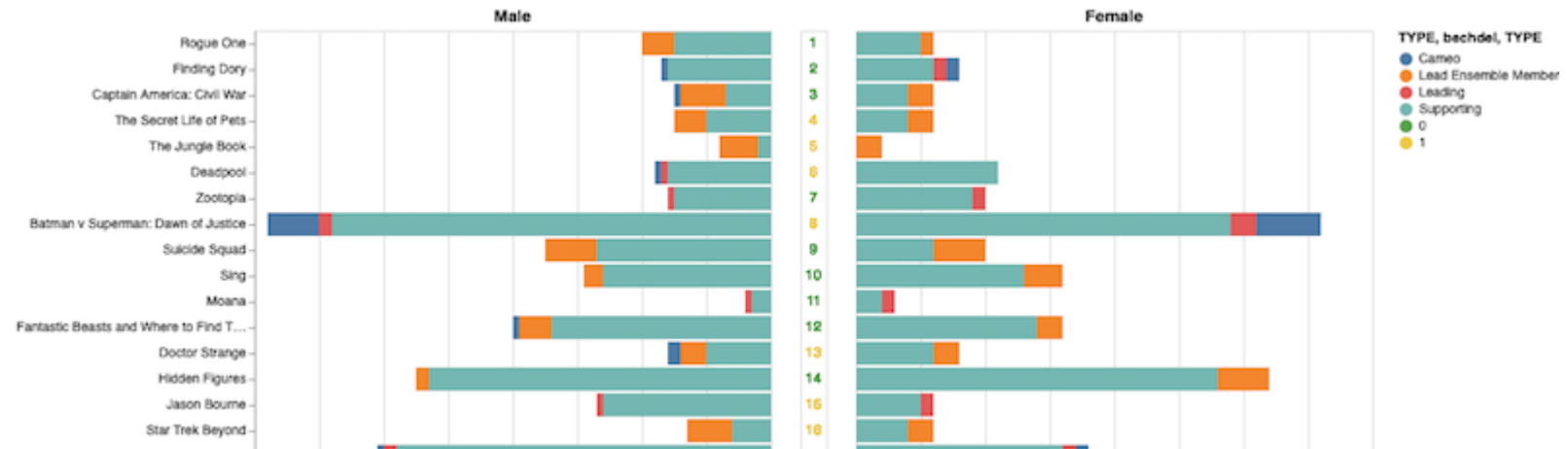


In [19]: *#hidden tests are within this cell*

## 2.5 Variables encoded 2 (5 points)

Complete the `variables_encoded_2` function. Return the number of variables encoded in the following plot (again, something like `return 5000; ...` we're using this for automated testing). If you have been able to complete the previous examples, the plot should look like this

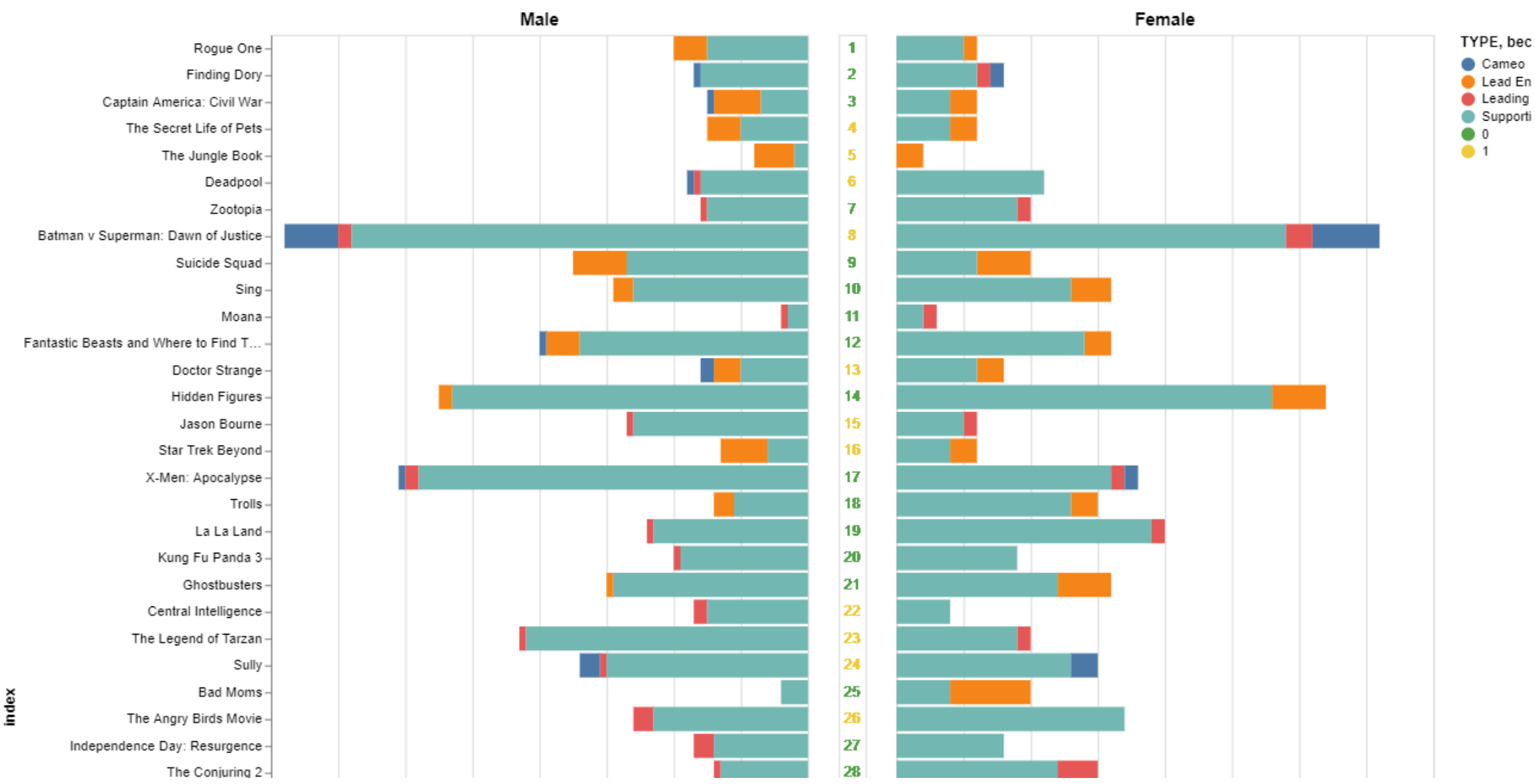


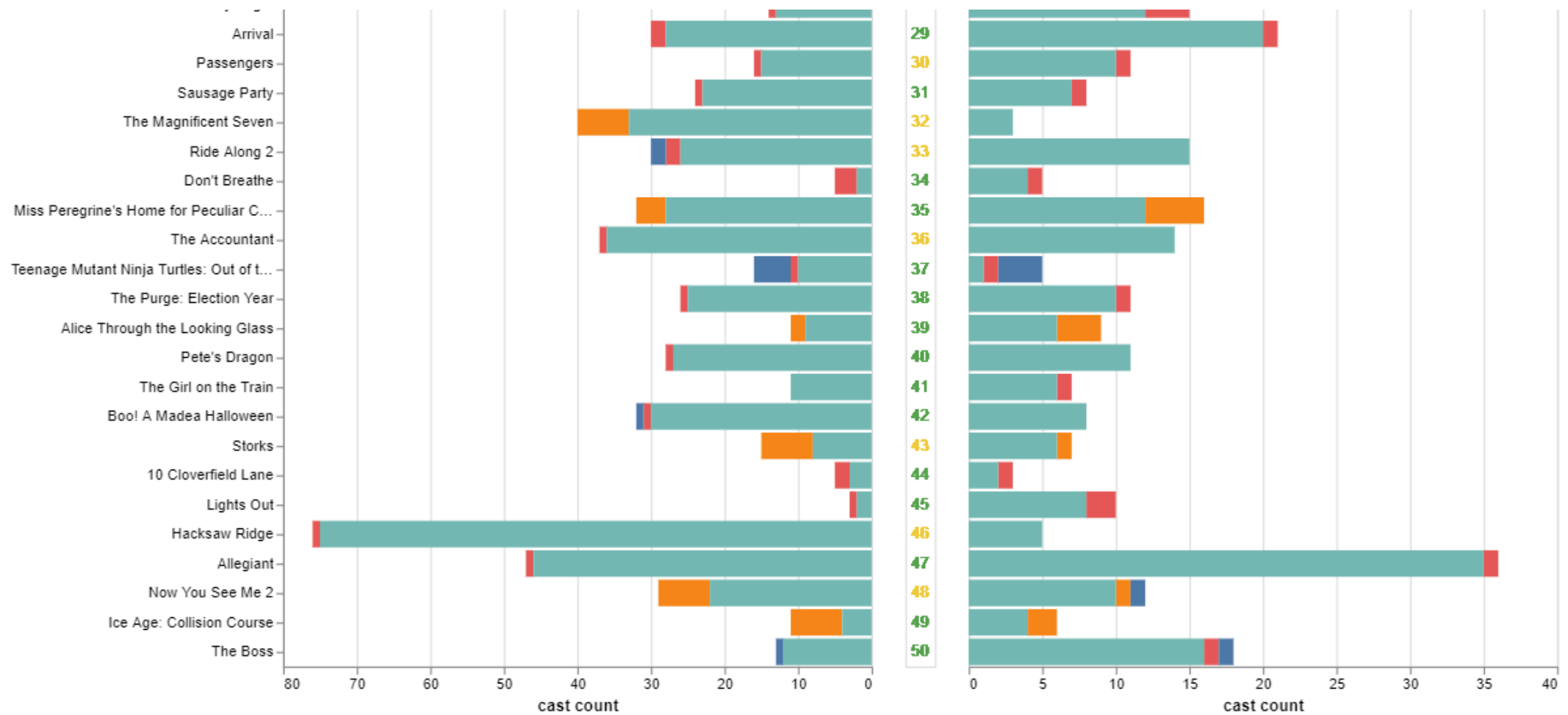


click [here \(assets/visualization\\_fullSize.png\)](#) to see the full-sized image.

```
In [20]: middle = base.encode(  
    y=alt.Y('Rank:O', axis=None), #1, 2 (ordinal desgination)  
    text=alt.Text('Rank:Q'), #3, 4 (quantitative designation)  
    color=alt.Color('bechdel:N') #5, 6 (nominal designation)  
).mark_text().properties(width=20)  
  
# merge together the three charts, male, middle, female  
male | middle | female
```

Out[20]:





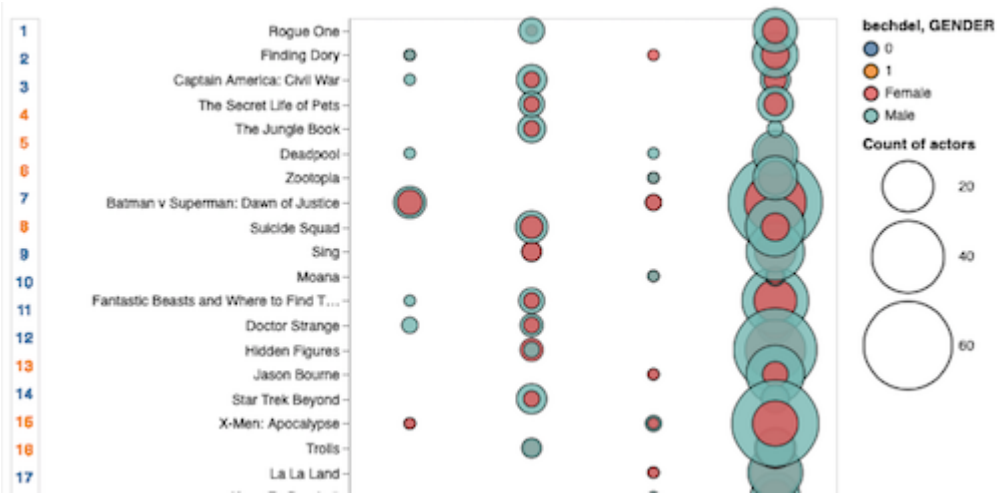
```
In [21]: def variables_encoded_2():
        """
        return the number of variables encoded in the given example
        """
        return 19
        #raise NotImplementedError()
```

```
In [22]: #hidden tests are within this cell
```

## 2.6 Alternative encoding 1 (20 Points)

Create a new visualization within the `alternative_encoding_one` function with the following encoding:

- Use circles as the mark
- Use the scale of the circles to encode the number of actors on each category
- Use the y position of the circle to encode the movie
- Use the x position of the circle to encode the type of actor
- Use the color of the circle to encode the gender of the actor
- Match the styling of the example



click [here \(assets/visualization\\_2\\_fullSize.png\)](#) to see the full-sized image.

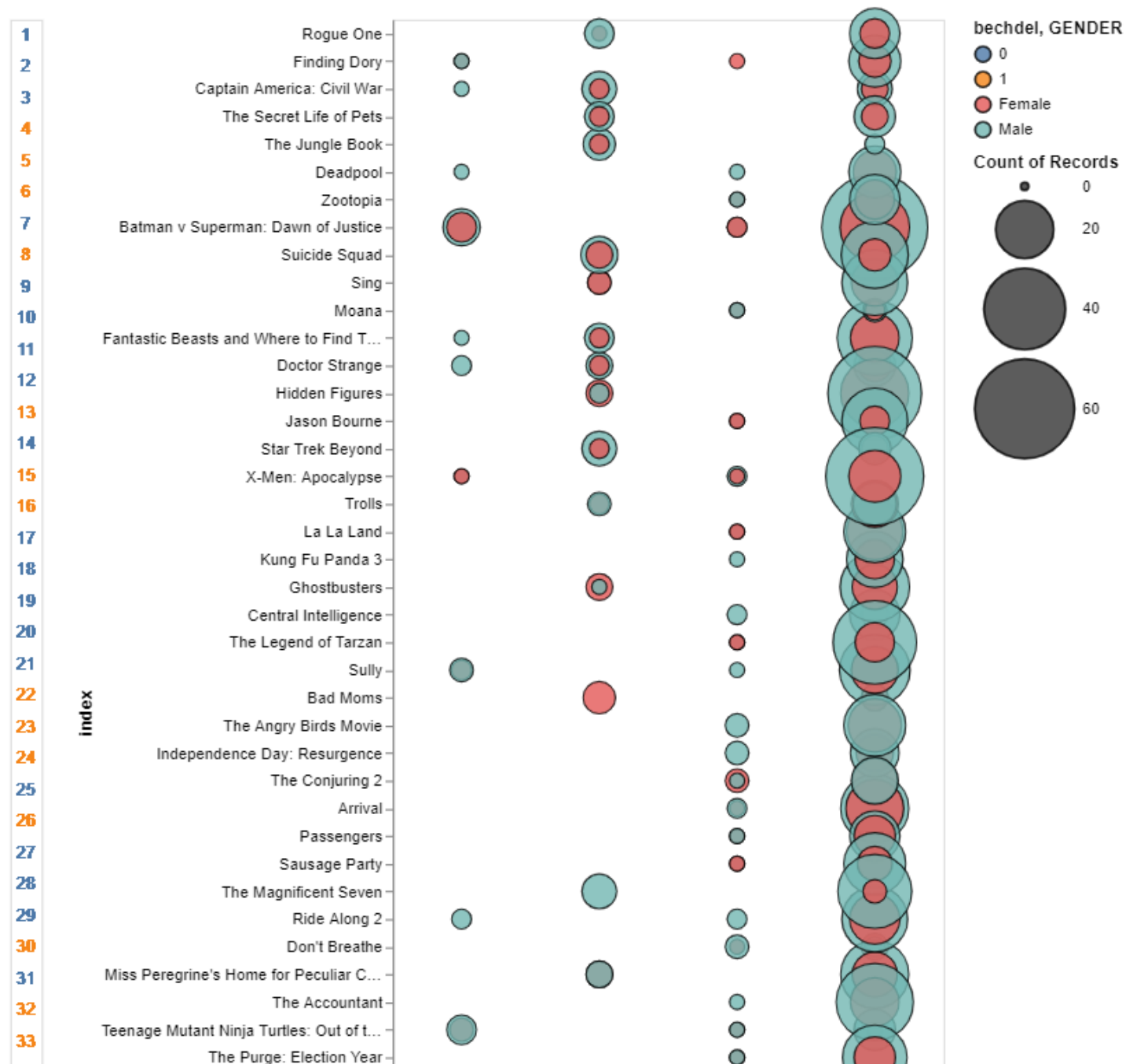
- *Partial credit can be granted for each visualization (up to 5 points) if you provide a description of what the missing piece of the function is supposed to do*

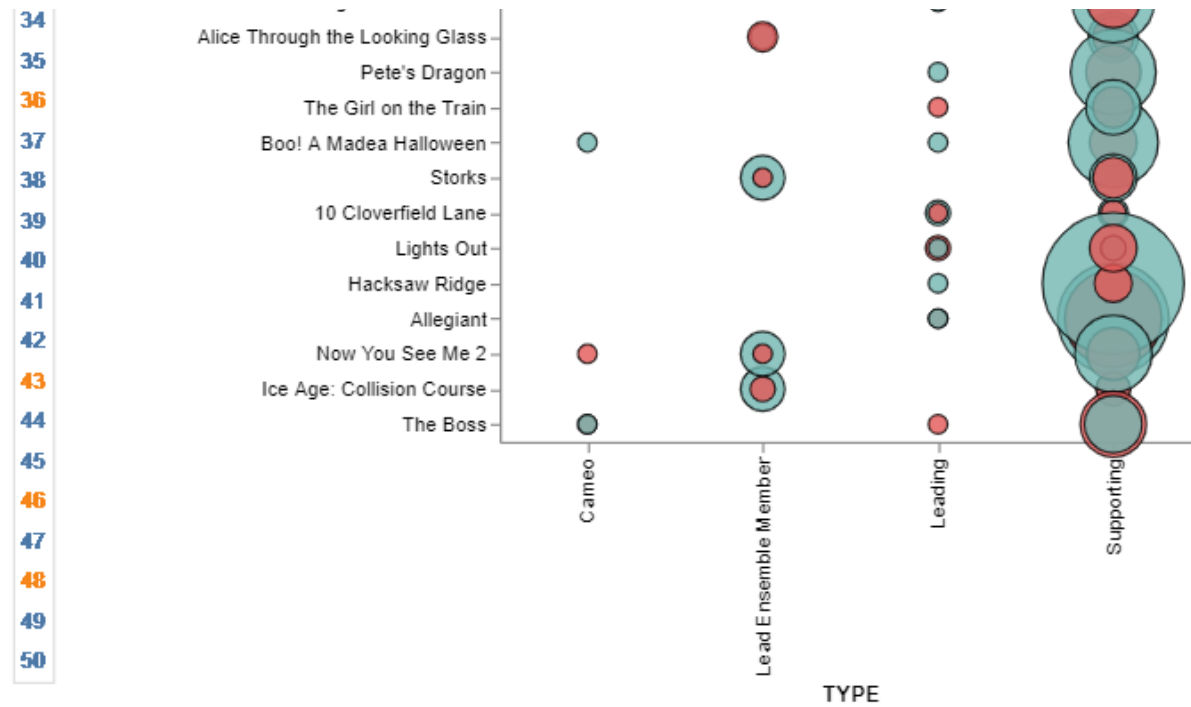
```
In [23]: def alternative_encoding_one():
        """
        return call to altair function for the new visualization
        """
        plot = base.mark_circle(
            opacity=0.8,
            stroke='black',
            strokeWidth=1,
            filled = True
        ).encode(
            alt.Y('index:N',
                  sort= movies_order),
            alt.X('TYPE:N'),
            color = alt.Color('GENDER'),
            size = alt.Size('count(GENDER)', scale = alt.Scale(range=[25, 5000]))
        ).properties(
            width=350,
            height=880
        )

        #raise NotImplementedError()
        return plot
```

```
In [24]: al_enc_one = alternative_encoding_one()
middle | al_enc_one
```

Out[24]:





In [25]: *#hidden tests are within this cell*

## 2.7 Alternative encoding 2 (25 Points)

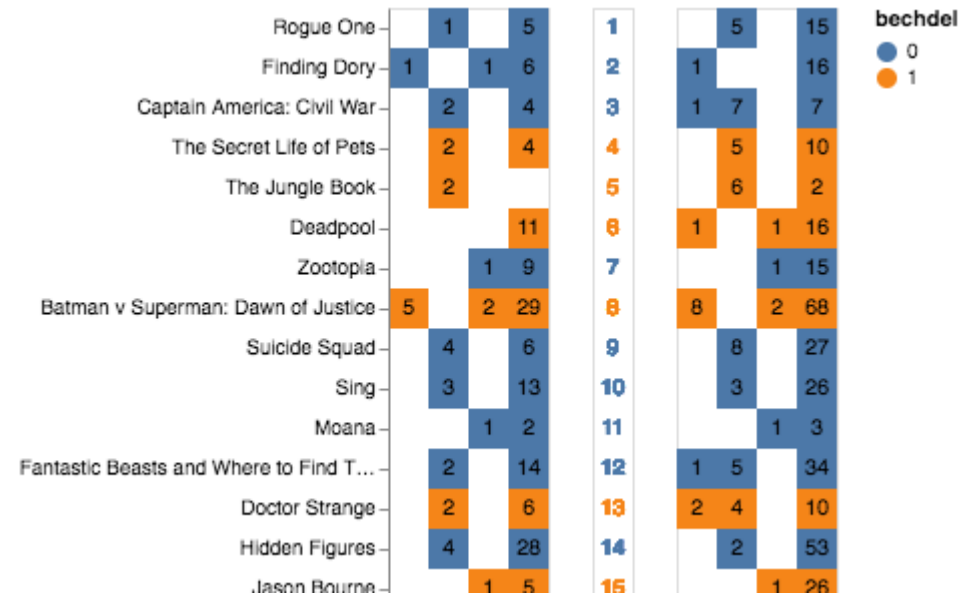
complete `female_actors_1()` and `male_actors_1()` functions to create a new visualization with the following encoding:

- The left and right plot should filter male and female actors respectively
- Use rectangles as the mark
- Use the text inside each rectangle to encode the count of actors one each category (gender, type and movie)
- Use the y position of the rectangle to encode the movie
- Use the x position of the rectangle to encode the type of actor
- Use the color of the rectangle to encode whether that movie passes the Bechdel test or not (bechdel variable)
- Note that only the female vis has labels on the left, the male vis does not

The top of the female plot would look like this (click [here \(assets/visualization\\_3\\_fullSize.png\)](#) to see the full-sized image):



If you've done everything correctly, your final visualization should look like the one below (click [here \(assets/visualization\\_3\\_full.png\)](#) for the full plot).



- Partial credit can be granted for each visualization (up to 4 points for each function) if you provide a description of what the missing piece of the function is supposed to do without need for an Altair working version



```
In [26]: def female_actors_1():
        """
        return call to altair function for the new visualization
        """

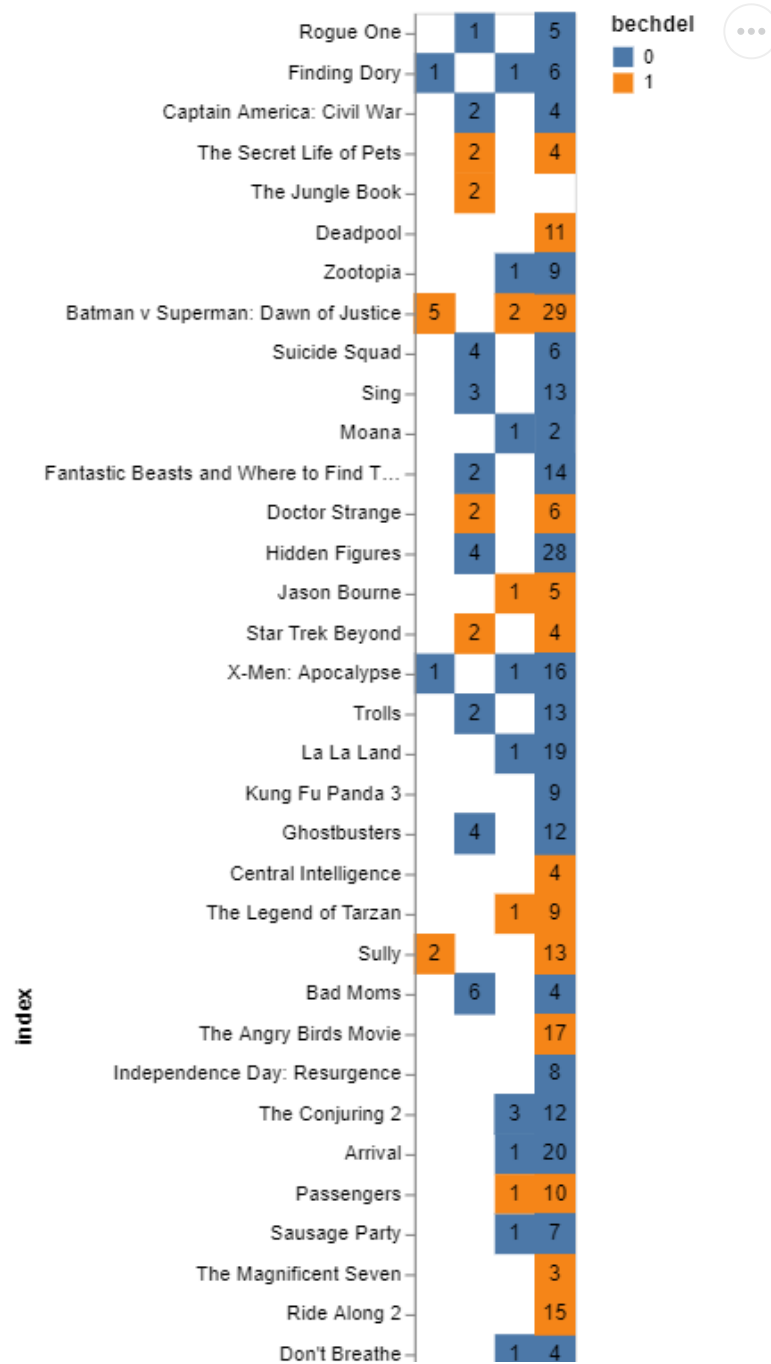
        plot = base.transform_filter(
            alt.datum.GENDER == 'Female'
        ).mark_rect().encode(
            alt.X('TYPE:N'),
            alt.Y('index:N',
                sort= movies_order),
            color = alt.Color('bechdel:N')
        )

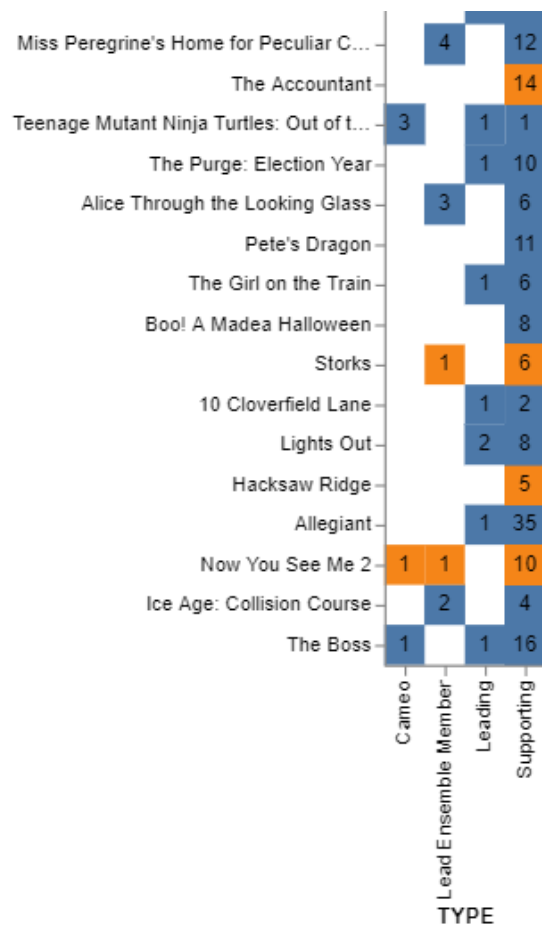
        text = base.transform_filter(
            alt.datum.GENDER == 'Female'
        ).mark_text(baseline='middle').encode(
            x='TYPE:O',
            y= alt.Y(
                'index:O',
                sort= movies_order,
                axis=None
            ),
            text = 'count(GENDER)'
        )

        #raise NotImplementedError()
        return plot + text
```

```
In [27]: f_a_1 = female_actors_1()
f_a_1
```

Out[27]:





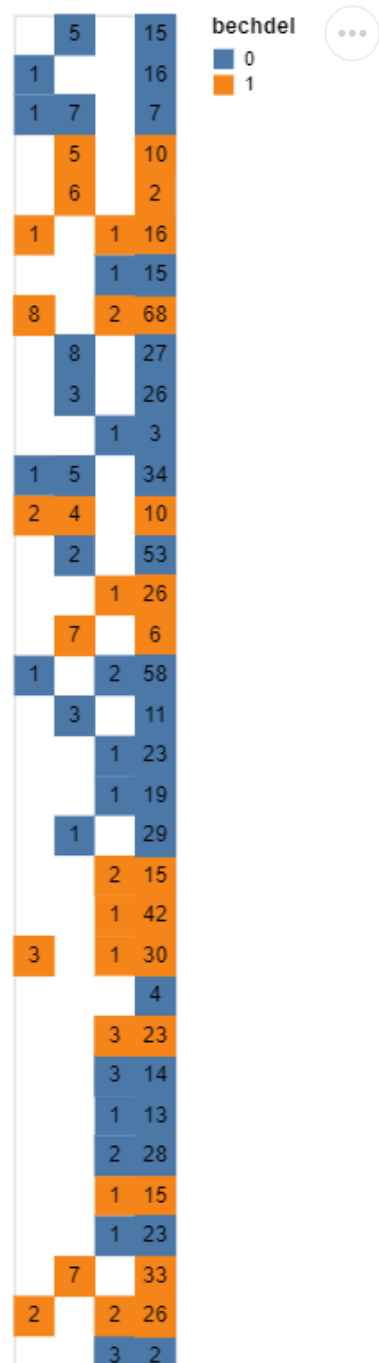
In [28]: *#hidden tests are within this cell*

```
In [29]: def male_actors_1():
        """
        return call to altair function for the new visualization
        """
        plot = base.transform_filter(
            alt.datum.GENDER == 'Male'
        ).mark_rect().encode(
            alt.X('TYPE:N'),
            alt.Y('index:N',
                sort= movies_order,
                axis = None),
            color = alt.Color('bechdel:N')
        )

        text = base.transform_filter(
            alt.datum.GENDER == 'Male'
        ).mark_text(baseline='middle').encode(
            x='TYPE:O',
            y= alt.Y(
                'index:O',
                sort= movies_order,
                axis=None
            ),
            text = 'count(GENDER)'
        )
        #raise NotImplementedError()
        return plot + text
```

```
In [30]: m_a_1 = male_actors_1()  
m_a_1
```

Out[30]:

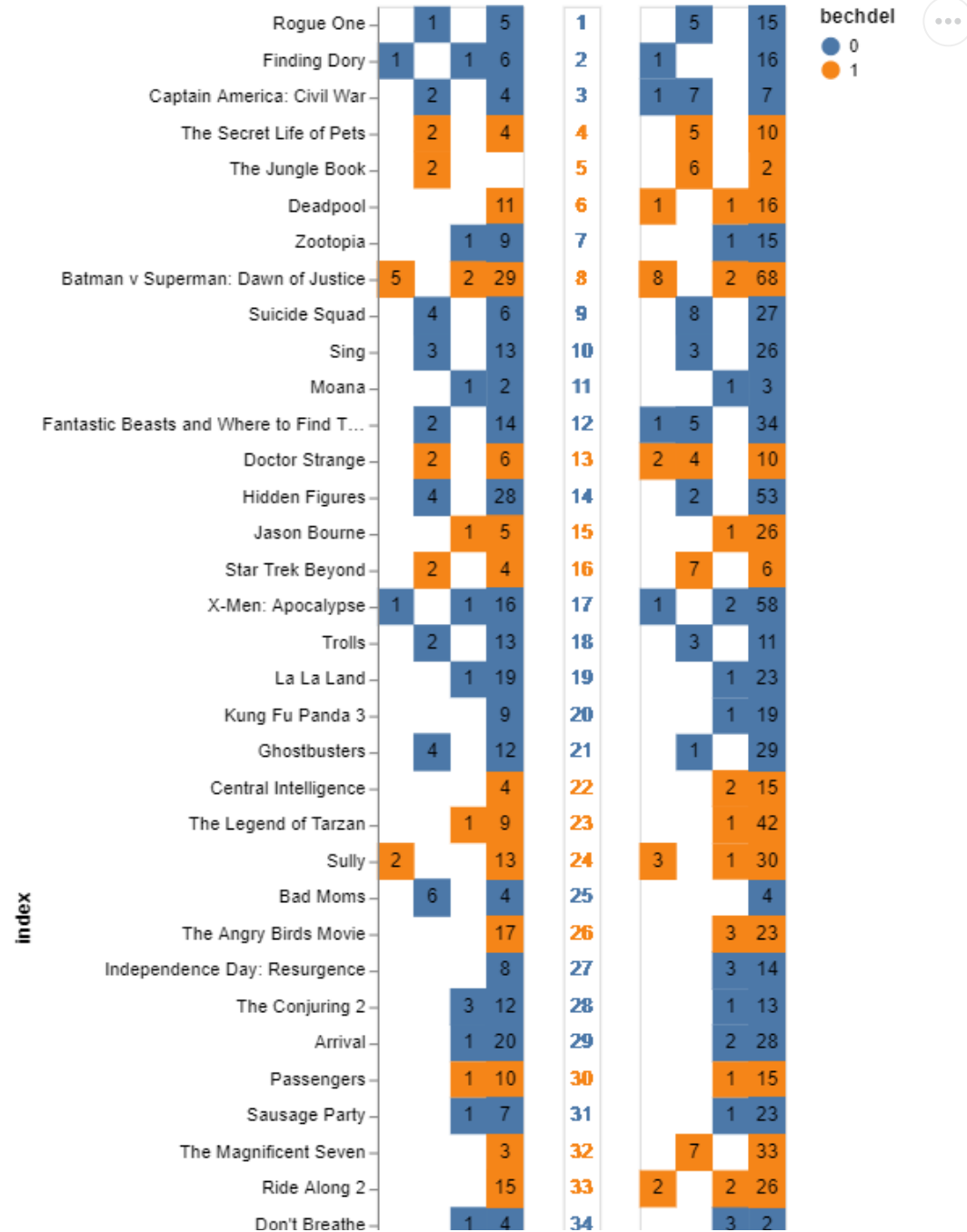


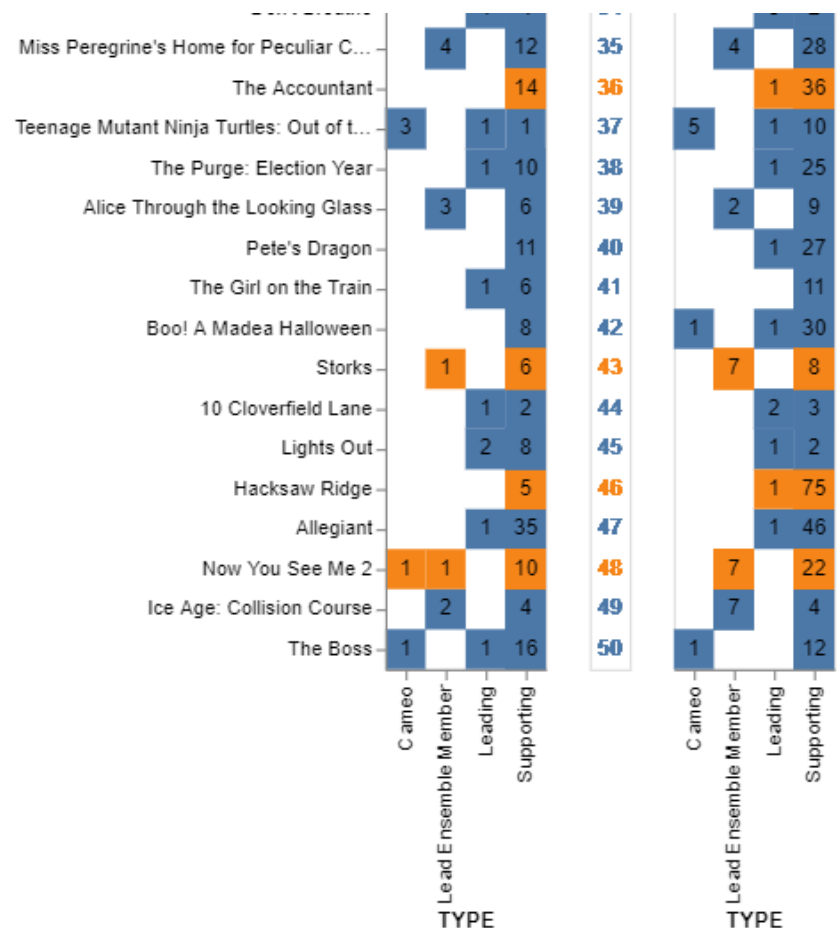
	4		28
		1	36
5		1	10
		1	25
	2		9
		1	27
			11
1		1	30
	7		8
		2	3
		1	2
		1	75
		1	46
	7		22
	7		4
1			12
Cameo	Lead Ensemble Member	Leading	Supporting
TYPE			

In [31]: *#hidden tests are within this cell*

```
In [32]: # create the visualization
f_a_1 | middle | m_a_1
```

Out[32]:





## 2.8 (Bonus) Compare effectiveness (5 points)

Look at the visualization for question 2.7. How does this visualization compare in terms of effectiveness to the visualizations in questions 2.5 and 2.6?

Visualization 2.7 would be the most effective visualization when compared to 2.5 and 2.6. While all three of these plots have the same level of expressiveness, 2.7 has a much more easily digestible display as a viewer could simply search for a movie and have a clear number displayed for the number of Leading, Supporting, etc. men and women in the movie. For visualization 2.5, the bar chart is also digestible, but there can be some room for error when attempting to discern whether the length corresponds to perhaps 26 or 27. Similarly, the shapes in 2.6 make it very tough to get a clear count of the number of actors in a particular role or movie. The one advantage of 2.5 and 2.6 over 2.7 however is that these visualizations have clear "Male" and "Female" titles. Visualization 2.7 would require an educated guess in order to decipher whether the figure corresponds to either a Male or Female breakout.



In [ ]: