Jeffrey Stern - jps394 | Jake Wright - jsw299 | Steven Li - ml653
INFO 3300 P2 | Write Up

Bubbles are Beautiful - An Analysis and Overview of the Tech IPO Market

A. DESCRIPTION OF THE DATA

Our data was pulled from the staple financial resources - namely *Bloomberg*, *S&P Capital IQ*, and *Reuters*. The larger majority though came from CapIQ. CapIQ (with a capable account → we accessed our account through the Johnson Graduate School of Management) has a variety of screening functionality which allowed us to tailor our search for the data that we were looking for. Specifically, our screen pulled over 4000 international IPO offerings (within the Technology sector) with information from 1995 to the most recent available date.[1] Each transaction contained a plethora of information about it including:

- Company IPOing
- Filing/Announcement date
- Exchange Ticker
- Sector, Group and Industry of the company
- A description of the company's business
- Company website
- Company's Country
- Offer Date
- Offer Stock Price
- Offer Market Capitalization
- Market Capitalization as of the date of the last pull
- Company's stock return
- CAGR (Compounded Annual Growth Rate)
- Stock Price data

However, this is not what we initially started with. Our screening evolved overtime as we went through the thought process of figuring out which information we wanted to plot. We knew we wanted to focus on IPO transactions but the amount of information available on IPOs through Bloomberg and CapIQ is overwhelming and so frankly we didn't start in the right place. Initially we grabbed how much money a company was raising during its IPO rather than the market valuation of that company. Ultimately we decided to go with market capitalization as the main metric.

[1] April 12th, 2016 was initial screening date

The stock price data includes a stock price for each respective company for every single week since 1995 meaning there's roughly 1000 data points for each transaction *(we choose weekly stock prices rather than daily because Excel simply couldn't handle the scale of the CapIQ screen with that many values to update)* meaning our initial screen provided us with > 4,000,000 data points!

We figured that it would be hard to visually represent and gain insightful takeaways from this amount of data so we decided to tailor the results. First off, we got rid of all the pulled transactions that were cancelled, withdrawn or suspended eliminating roughly 700 transactions. Second, we had to go through the remaining transactions and get rid of all the ones with faulty information (meaning CapIQ and/or Bloomberg didn't have the proper data points for a specific category (i.e. offer valuation or offer date)). Next we decided to focus solely on the United States leaving us with the domestic companies that are displayed in our visualization. We felt that this amount of filtering and subsequent transactions (in the 100's rather than 1000's) made sense to work with (still >300,000 data points though!).

→ That was how we approached filtering the data. The circle visualizations were taking from individual transactions and the subsequent takeaway graphs came from calculated aggregate data from those individual transactions.

B. MAPPING FROM DATA TO VISUAL ELEMENTS

i. Scales

Now that we finally had accessible data in the form of CSVs, we were ready to get our feet wet with visualization. For the most part, this was pretty straight forward. We used four scales, three of which are linear and one of which is ordinal. They are as follow:

1. Y-Scale (Linear)
    a. This scale creates a y-axis that generates off of time that's passed since 1995. The domain for this function is date's from 01-01-95 to 12/31/15, and ranges from the height minus the padding to the padding (so that the y-axis is in the correct direction).
2. Radius Scale (Linear)
    a. This scale was created to plot linearly increasing radius size of the nodes as correlated with the initial market value. I.e. We planned on having larger companies appear with larger radii (and vice-versa) based off a linear correlation to the size of the initial market value.
    b. This scale takes the minimum and maximum Market Values as a domain on a linear scale, and the range for this scale is [30-200] because we determined that these provide good sizes for the space and size of the SVG we are working with for our visualization.
3. Color Scale (ordinal)
    a. The color scale is used to visually create groups for the different industries of tech companies that we are displaying. This is an ordinal scale and because there are 9 different industries, I set the domain to include the numbers from 1 to 9, and then set the range to include 9 unique colors (listed below for reference):
        i. .range(["#F49AC2", "#03C03C", "#779ECB", "#FFDF00", "#CFCFC4", "#FF6961", "#FFB347", "#CB99C9", "#1FCECB"]); //magenta, green, blue, yellow, orange, red, violet, grey,
4. Text Length Scale (linear)
    a. This final scale is a linear scale that determines the maximum length a company's name can be before its name will overrun the boundaries of a circle. Using this as a boolean check, to see if the particular node's name fits within the size of its circle, we can determine the companies whose names do not fit and as a result we can determine which companies need to have their stock ticker displayed instead so that the description will fit within the circle

b. The domain for this scale was [30-200], which is equivalent to the range of radii, and the range for this function is [12,35] denoting that at radius 30 a circle can have at most 12 char's in its name before it uses its Ticker symbol, and that at radius 200 a circle can have at most 35 char's.

ii. Force Layout/Nodes

The force layout was used to populate the circles and allow for dynamic elements of our visualization such as the data collision. To create the layout, we first bound the data points that we chose to use to a d3 data selection called nodes. These nodes would serve as the primary means in which we would iterate through and interact with our visualization. The force layout allows us to create an onTick function that allows us to update the x and y fields of our nodes and circles. This onTick function is used to handle collisions of the nodes with the walls of our visualization as well as serve the all important purpose of limiting movement of circles along the y-axis. We initially implemented the force layout by binding the circles to their strict y position; however, given that a perfectly precise position was not necessary to the visualization and a more generalized placement more than provides enough information for the viewer, we allow for very limited movement along the y-axis. We found that this allows for many more companies to be comfortably shown on the visualization as circles can gently nudge others to make room a bit more room for itself. Furthermore, we also felt that having the circles collide would better allow for a cleaner look, and as such, we used and slightly modified the collide function provided on (http://bl.ocks.org/mbostock/1748247) as was able to provide most of the funtionality that we wanted.

iii. Circles

After the axes were up and running, and the Force Layout was conceived, we were finally able to visualize the data points we had. To do this, we mapped the y coordinates on a range (so that the nodes could interact with one another) according to the year that company IPO'ed, and allowed any x coordinate. Furthermore, depending on the data, we plotted the nodes with a certain size and color so that we could distinguish one data set for another. If we were mapping a particular industry we would use the ordinal colorScale. If we were mapping on a particular Market Value we would use the linear Radius Scale.

And then our group of 3300 said "Let there be data!" and there was, although it was chalk full of some errors that we had to address.

The first problem we noticed is that there were several large outliers (including Alibaba) that IPO’ed for a much larger value than the rest of the data set. These data points threw off the Radius Scales making it so there were few nodes with large radii and many nodes with small radii. As we would find later, reducing the set to just U.S. companies allowed us to find a happy balance in circle size. Instead of having a domain from [0, ~300,000MM] as we did with some of the outliers, once we changed out data set, the domain switched from [0, ~80M], which produce a scale 1/4 of the size and allowed some smaller companies to be represented more intelligently than just the minimum sized radius (30).

iv. Text

Appending text proved to be one of the more difficult aspects of the project. If we had more experience and IF WE COULD GO BACK and re-design the project, we would have implemented the .node class so that each text element/circle element and their data would and been accessible through an instance of a node.

However, this was not the case and we had to append a lot of the text manually (although it was definitely not hard coded). As a result, we had different classes of text elements. These were the .normalText and .uniqueText classes as described below.

1. .normalText:
    a. This class was used to describe text elements that provide the basic information for a given node, **specifically** its name and its initial market value. This is the standard text that cover the visualization and should cover the visualization unless turned off.
    b. We append these two fields to each node, and move these fields with onTick, so that whenever a node moves, its .normalText moves with it.
    c. On the mouse event, mouseenter for the nodes, we call:
    ```
            //currently gets rid of all text
        d3.selectAll(".normalText").attr("visibility","hidden");
    ```
        i. This hides all the text so it focuses on the node you are currently hovering over
    d. Furthermore, when the mousevent mouseleave takes place, we restore all this text
2. .uniqueText:
    a. For this class, we named it .uniqueText because there would only one instance of this element at a time: when hovering over a node, enlarge and append all the necessary information with it as the class .uniqueText

b. The data being relayed includes: Company Name, Total Return, Business Description. All of this information is being appended in the dispaly_data helper function.
c. This Data is displayed in an opposite fashion to .normalText; i.e. when the mouseenter event occurs, I call this function:

```
setTimeout(function(){if(mouseNode.style("opacity") == 1){
                    display_data(d);
               }}, 200)
```

   i. This purposefully displays the data of ONLY the node currently selected. We ran into some synchronization issues where first the nodes would flicker from their ON-OFF states, and next when we implemented a counter to handle the display_data, the data of other nodes would be displayed when you previously scrolled over them (because the counter would necessarily decrement below the specified value. But because only one node is selected at first and we change its opacity to 1, this function works.

d. Furthermore, on the mouseleave function for nodes we select all .uniqueText using d3 and remove it.

For all the text we ran into issues on making the text fit within the circle of a particular radius. To do this we used the max length Scale as described above, we eventually had to manipulate our code to account for the corner cases and make sure that no text extends beyond the boundaries of its node. In implementing this, we would set the attributes x and y equal to the nodes value for x and y and then afterwards we would manipulate its position using the dy attribute. THis allowed us to scale descriptions because all text/node/circle elements were plotted from the same spot in (x,y) coordinates.

v. Filters

We decided we wanted the graph to be more interactive, so we decided one could limit the data points and change the colors based off of a number of different factors. As such we ended up implementing two different filters.

1. Industry Filter
   a. The industry filters allows the user to exclude/include data based off of a particular industry. To create this filter, we append a form with labels (whose

data are based off the 9 different industries within the tech sector) all with checkboxes

   b. Initially they are all set to true (i.e. they are checked)
   c. We created further functionality, based off of clicking on the filter, that updates a loop that checks to see which of the buttons are check. From this point, if the value of the label is true, we restore the original data to the nodes from that industry. If the value of the label is false, we change the node's from that industry so that their circles shrink and their text disappears.
   d. A frustrating bug that we ran into was in the implementation of reset of the nodes after filtering. What occurred was that after a node's size was decreased, mousing over any other node would reset all the text sizes. When attempting to fix this bug, we wanted to access an attribute of the .node class; however, we found out that unlike jQuery, d3 does not have any function to access parents and the only way to circumvent this is to use html dom functions and reseting a new d3 selection for the parent.

2. Color Filter
   a. The color filter was designed to change the colors of the nodes based off of different properties. The initial design when you load the page is that the colors are split using the color scale as described before to show companies by a particular industry. However, we thought it would be cool to view the nodes by Profits/Losses and as such we created this second filter to make this an option of choosing between these two states.
   b. Now, when selecting "Total Return" the colors of the nodes change to green/red when they have been positive/negative total return since their IPO.
   c. This filter is very similar to above but instead of using checkboxes, we used radio buttons so that each value is mutually exclusive (i.e.: only one can be true at a time).
   d. Furthermore, we use an onclick event that allows us to check the values of the radio buttons and depending on which is true, we give them a set of particular colors. This filter was in some ways more simple because we only modify the COLOR of the nodes, not the radius of the circles or the elements appended to them (i.e. making the text disappear). As such we did not run into the bug that we found when making the industry filter.

vi. Helper Functions

Finally, we used several helper functions that allowed us to model the data; we found that in many aspect of the project we were writing similar code, so we tried to condense it so that we could create a sense of abstraction and know exactly what our code is trying to accomplish by using helper functions as often as possible.

1. greenOrRed:
    a. This is a generic function that takes a node and determines from string splicing its data whether or not it has a positive or negative total return since its IPO. If it has a positive return this function returns “green,” if it has a negative return this function returns “red,” and if there is a neutral return this function returns “blue.”
        i. These strings are then integrated to be colors, or used as a value to determine whether or not the stock is positive/negative.
2. deleteAnd:
    a. This is a generic function that takes a string and removes ‘ and ’ and ‘,’ from that string and replaces those instances with a “/”. This is very valuable for minimizing space (think fitting text within the space of a circle) taken up by those extra characters. One place this function is used is to append the industries as .uniqueText to the nodes when they are being required to display its data. I.e. this is a helperfunction called within the display_data helper function.
3. Tokenize_dscription:
    a. This is a generic function that takes a business description as a string value (from a node) and tokenizes it, meaning it split its description into multiple lines so that it will fit inside circle. This function returns an array with the business description split up into different sections based off the fact that the maximum length of any given line of the business description has to be smaller than the charLimit, which is set in its own field. By setting a maximum distance for each line, we know that we can fit the entire business description within the circle.
4. Display_data:
    a. This function is somewhat the bread and butter of our data and interactivity. This function displays the data as text on the currently expanded node. It is called from the node.on(mouseenter) function with the set timeout, as shown before. The reason why this function works is because we can set all of the text as the same class, .uniqueText, and that way, all the information that is printed from this function can be removed in the instant when the mouse is no longer hovering on the nodes, which is what happens in the node.on(mouseleave) function, and as such all of our data is moved.

C. THE STORY:

The truth is that we went into this project without a story tell - we weren't looking for anything in particular beyond getting a better feel for the Technology IPO industry. This made it pretty easy for us to focus on the aspects of the project we found interesting and try to formalize some of our takeaways. Here are some of the things that we noticed:

With regard to IPO valuations overtime, it seems pretty apparent that IPO valuations are trending upwards which, to us, seemed representative of the "unicorn" mentality which has garnered significant media attention recently. While historically companies would go public much earlier in their respective life stages at much smaller valuations, more recently, technology companies are really extracting as much capital as they can from the private markets prior to IPOing leading to higher market capitalizations when they do go public. Some prevalent examples contributing to this Unicorn trend at the moment are private companies like Palantir, AirBNB and Uber which are all individually valued at more than $25 billion - if they were to go public tomorrow, they would be some of the largest IPOs in history. At the moment, there are more than private 161 private companies valued at more than $1 billion implying when they go public, they will certainly contribute to the upwards trajectory of tech IPO valuations.[2]

The turn of the century also led to some cool retrospective takeaways relating to the famed "Tech Bubble" (also because we had narrowed our data to only domestic companies). At the time, an enormous amount of companies, caught up in the hype and promise of the internet and stock market, tested the public markets which in retrospect lead to one of the largest stock disasters in recent memory. From our data, this was evidenced just by the sheer magnitude of companies that IPOd during this time relative to later times. More specifically, there was a pretty cool takeaway relating to a pretty nuanced part of the data, namely what exchange a company trades in. The household market exchanges in the US are the New York Stock Exchange (NYSE), the S&P 500, and the NASDAQ which each encompass a specific type of company (the NASDAQ is most frequently associated with technology companies). The OTC market though is an "over the counter" exchange qualified in contrast to the aforementioned formal exchanges. Typically a company will trade on the OTC exchange when it is still too small from a valuation standpoint to meet the formal exchanges' listing requirements. What is interesting here is the number of OTC offerings during the Tech Bubble relative to any other time. Hindsight is 20-20 but such a spike in OTC company listings around the turn of the century, which are typically not the most financially sound and which are often penny stocks, could have sounded the alarm to investors that something odd was occurring.

Finally, one other thing we'd like to highlight is that there actually still have not been any tech IPOs in 2016[3], which relates back to the aforementioned Unicorn market and is further

[2] https://www.cbinsights.com/research-unicorn-companies
[3] arguably one IPO if you count Dell's SecureWorks' spinoff

indication that technology companies are staying private longer leading to higher valuations when they go public.