The Holy Grail of Venture Capital
e^x Team: Circle, Julian, Nitin, Mudit, Thomas
Final Report

In venture capital and private equity, predicting the success of a start-up is as much a science as it is an art. Sure business plans, customer metrics and financial models can help and the size of the addressable market as well as the product market-fit are key success indicators. But early stage companies don't have those; all they have is an idea and founders planning to execute upon it.
So how can we tell whether a company will be successful or not at such an early stage? By nature, it is a guess. However, when millions of dollars are at stake, it is crucial for our guess to be as educated as possible. This is what we are trying to achieve with e^x: smarter investments, exponentially. We use data science to provide investors with additional insight to help them predict the success of a founder. However, as we started ideating about our solution we were faced with a problem: How do we define success? We defined our success metric through the numerous talks we had with venture capitalists:the increase in valuation from from Series A to Series B. We were ambitious and motivated and we were aiming to build a regression model, predicting that valuation increase based on founder/CEO data. However, as we got going, we pivoted, instead exploring the most important features that are present in a successful founder and building a similarity calculator.

We cleaned up our data and performed feature engineering using Pandas. First, we removed any founders that did not have a valid value for the valuation increase (either the value was a NaN or we did not have information on that founder). This would allow us to analyze only those founders that had a change in valuation, better or worse, so that we could stay true to our definition of success - this increase in valuation across funding rounds. Next, we added features such as "Ivy League School" and "Bay Area School" based off of the data we had already to see if there was a correlation between a certain type of school and valuation increase. We also standardized some values such as major so that entries like "Chemical Engineering" or "Civil Engineering" would be turned into just "Engineering". We did this because we did not have enough data points with a specific major to gain insightful info so we grouped majors together.

Our solution basically came down to taking a weighted sum of all the feature contributions to the similarity score and then normalizing it. So the problem came down to:

- How are we going to determine the weights?
- How are we going to determine the contribution of each feature?

For the weights, we decided to make use of the fact that Random Forest Regressors determine the information gain of each feature to decide the branching. So, we trained a Random Forest Regressor using our features values as input and series valuation increase as output in the hope of seeing how "important" or "relevant" each feature is to a valuation increase in the company. In doing so, we are essentially scaling our features by how big of a contribution it makes to the success of the startup. From this process, we get weights that all sum to 1, which is essential in our next step.

After getting the weight of each feature, we needed to determine how to find the contribution of each feature to the similarity score. We decided to use an inverse absolute difference loss function to penalize dissimilarities, but much more smoother than a Dirac delta function. Once we computed this soft penalty, we scaled it by the weight from the Random Forest Regressor and added it to the contribution.

Notice that throughout this entire process, all of our values are normalized between 0 and 1 (both the weights and the soft penalty). This ensures that we are comparing apples to apples and not including any bias in our similarity calculations, though this is something that we could have definitely taken into

account if we had more time. After computing the sum, we normalized the score by the sum of the weights, and this result is guaranteed to be between 0 and 1 because of the fact that all our values are between 0 and 1.

So, the algorithm essentially computes the similarity score between the founder in question and all of the founders in our database and outputs the k (that we decide) “most similar” founders, along with their similarity score and other information about them.

We collaborated with Shomit Ghose of Onset Ventures, Pradip Shankar of Ericcson Strategic Investments, Paul Arnold of Switch Ventures, and other VC’s in order to understand how VC’s determine which ventures to invest in and how they define success. Our mentors were very helpful for the most part because they assisted us in narrowing our focus so that we can just look at relevant information for founders. Many VC’s had similar sentiments in that hoped we could somehow quantify the personality of each founder. This guided us towards looking at more personality traits of the founders and ultimately decided which characteristics we were going to analyze. Our main mentor, Patrick Chung, was not as helpful as the other VC’s that we reached out to.

Our project certainly has limitations. We only had limited data sources, especially for our dependent variable: the valuation increase which we only found on pitchbook. The nature of the project also limits the data we could gather: every piece of data must be valid at the time a specific founder started his/her company. For example, the number of facebook friends is time dependent because the one we could get is the current number of friends in Sep, 2017.

Besides, these websites’ legal requirements also prevented us from using web scrapers. The team had to manually input all of the data by hand.

Lastly, some biases are included in this project. Since it is natural for people to hide their failure history, information in LinkedIn could be biased towards successful ventures. The data in Pitchbook and Crunchbase also includes more successful stories than failed ones. Finally, the VC’s biases are passed on to our project. Most of the successful stories are the outcomes of beliefs and investment ‘thesis’, skewing already who and who is not successful.

Moving on, we would focus on developing a scalable and efficient data collection process. This would allow for more features and more founders, leading to better insights. Finally, we would work on building a robust architecture, including a database management system, a validated user interface and an automatic update cycle for our RFR.

Though everyone had a say and worked together on everything, throughout the project Nitin was in charge of Data processing and feature engineering as well as building the front end. Mudit focused on data collection and external outreach. Circle worked on data collection and keeping track of our project’s limitations. Julian coded the back-end and translated the similarity algorithm into python. Finally, Thomas worked on data collection, developing the similarity algorithm on paper and extracting the necessary weights.