Valuation Growth Using Deep Learning Methods

Mouhssine Rifaki[1]

Abstract

Venture capital plays an instrumental role in the modern American economy. Corporations such as Google, Apple, Facebook, Instagram, PayPal, Tesla, SpaceX, Airbnb, FedEx, and Intel all received venture funding to grow into the iconic companies they are today. As of 2015, venture-backed companies made up 17% of U.S. public companies, accounted for 44% of R&D spending, and employed 11% of U.S. citizens [1]. Despite the unequivocal impact venture capital has on the United States, the process behind venture investment decisions remains manual, subjective, and unsystematic.

This paper explores whether venture capital can benefit from machine learning, particularly deep learning, to make investment decisions in a more systematic and scientific manner. Specifically, we aim to predict the valuation step-up multiple in the subsequent financing round of venture-backed U.S. companies. The primary dataset for our model comes from Pitchbook, a popular commercial dataset of company and funding information. We produced a regression-based model using a fully-connected 10-layer neural network to encode features and predict the valuation step-up multiple. Results show that for the regression task of predicting company valuation step-up, deep learning techniques meaningfully outperform statistical inference models such as linear regression. However, non-deep learning models, such as random forest, appear to be better suited for this regression analysis.

Keywords

fully-connected neural network, venture capital, regression, company valuation prediction

[1]**Paris Dauphine University - PSL**, *email: mouhssine.rifaki@psl.eu*

1. Introduction

Venture capital is a type of financing where investors provide capital to startups to finance growth in return for equity. After interviewing over 20 investors from top venture funds such as Bessemer Venture Partners, Andreessen Horowitz, Sequoia Capital, Redpoint Ventures, and GV, we have defined the typical early-stage venture capital process as follows: investors conduct research on the team, market, product, competitive landscape, and financials. The investing team amalgamates research and subjectively weighs data points to arrive at a binary decision of whether to invest or pass on the opportunity.

A successful investment is one where the company's valuation appreciates and the investor is able to liquidate their stake at the heightened valuation, generating a profit. Given this, the ideal task is predicting the valuation multiple at the time of exit. However, potential exit and liquidation events are unpredictable, vary significantly in time horizon, and are often not documented. Therefore, we defined our goal as predicting the valuation multiple in the subsequent round of funding. While we recognize that gains cannot necessarily be realized in successive rounds, the valuation step-up multiple is a key indicator of future exit value.

We limited our dataset to venture-backed companies in the U.S. that raised a seed round of financing and were founded between the years 1990 and 2020. By leveraging Pitchbook data, a popular commercial dataset of company and funding information, we obtained over 160 features, ranging from founder name (from which we could deduce gender) to `industry_code` (from which we could extract market-wide growth rates). After extensive cleaning, pre-processing, and transformations, we refined our dataset to 30 core features. These features were fed into a custom-trained, 10-layer dense, fully-connected neural network that encodes the features and predicts a numerical valuation step-up multiple.

2. Related Work

Some previous work has attempted to predict the success of a startup based on information regarding the company and its founders. One notable project, titled **The Holy Grail of Venture Capital** [2], was conducted by a technical and VC-experienced team from the University of California at Berkeley. This team utilized traits of founders, such as fear of failure, persistence, persuasiveness, reliability, competitiveness, network strength, and trust, to determine the likelihood of success. These traits were scaled as quantitative measures

from 1 to 5. Using data from Crunchbase and a founder survey, the team processed eight features and one target output after extensive pre-processing. For their model, they compared algorithms including logistic regression, SVM, perceptron, Naive Bayes, XGBoost, and random forest. Despite the comprehensive comparison, they did not explore using deep learning techniques.

Another prior study was conducted by Will Gornell and Ilya Strebulaev, who developed a valuation model for venture capital-backed companies with multiple rounds of financing [3]. Their model utilized data from Pitchbook and Genesis, focusing on U.S. companies from 2004 onwards. The dataset included approximately 19,000 companies and 37,000 financing rounds. The paper explored regression models to assess how current value, value change, and prior contractual terms influence the terms of a new financing round. Their results highlighted an overestimation of post-valuation figures but aligned with price reports from finance intermediaries in the VC industry.

3. Dataset and Features

Our research utilized data from Pitchbook, one of the premier commercial datasets of startups and venture capitalists. We began our data collection by scraping every company and fundraising round from the Pitchbook database. We restricted our dataset to U.S. companies that had raised a seed round of venture financing and were founded between 1990 and 2020.

While the raw dataset included over 10,000 rows, a large percentage of entries were sparse and included duplicates. To address this, we mapped every company and fundraising round to a unique company identifier and deal identifier, respectively. Duplicate entries were removed using the unique deal identifier, and companies without consistent round-to-round information were excluded. After data cleaning, our final dataset contained 34,265 unique deals across 22,067 unique companies.

Given that we aim to predict the valuation step-up multiple in subsequent rounds, we ensured our data was well-distributed among various initial fundraising rounds. Figure 2 illustrates the distribution of rounds across the five predominant fundraising stages.

An important factor to address in our dataset was survivorship bias, given the high failure rate of startups, especially early in their development. Our dataset inherently favored successful companies, as it was limited to those tracked by Pitchbook. According to the National Venture Capital Association, roughly 25% of venture-backed businesses fail, whereas only 7% of companies in our dataset had failed—a notable disparity [1]. To mitigate this bias, we took a longitudinal approach to data collection and factored company status into our predictive model. Specifically, companies labeled as "Out of Business" were assigned a valuation multiple of 0 in their final round. While these measures helped reduce survivorship bias, it remains a limitation of our model.

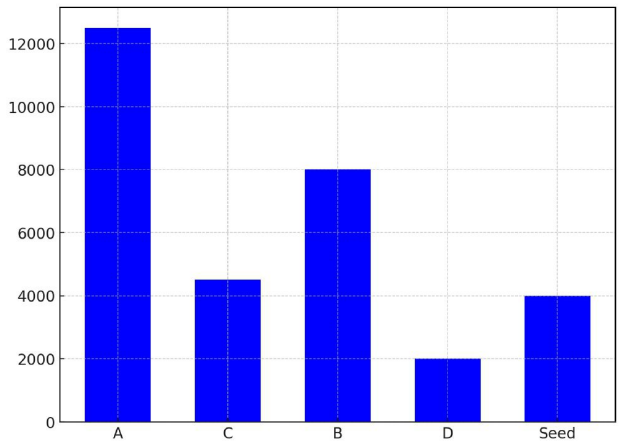

Figure 1. Distribution by fundraising round (Seed, A, B, C, D) in dataset.

3.1 Data Processing and Transformations

The raw dataset included over 160 columns, such as founder demographics, company information (e.g., year founded, location, employee count), customer count, number of deals, and features of the fundraising process (e.g., deal size and liquidation preference). A complete list of raw columns is available in the appendix.

We performed extensive pre-processing and transformations to derive meaningful features. Table 1 outlines the primary pre-processing and transformation techniques employed.

To standardize values over time, we integrated a GDP-deflator. We also applied normalization to features spanning multiple ranges to constrain values to a common scale. Normalization shortened the time to model convergence, as gradients reached local minima more efficiently.

3.2 Generating the Output Variable

The dataset did not natively include our desired output variable: valuation step-up multiple. We calculated this label by grouping all rows by `company_id`, iterating over each deal, and applying the formula:

$$\textit{valuation step-up multiple} = \frac{\textit{current round valuation}}{\textit{previous round valuation}} \quad (1)$$

For example, if the post-money valuation of a Seed round was \$10M and the post-money valuation of the subsequent Series A round was \$20M, the valuation step-up multiple would be

Category	Pre-processing Techniques
As-is	"employee_count", "year_founded", "deal_number", "percent_owned", "percent_acquired", "pre_valuation", "raised_to_date", "deal_size", "price_per_share_x", "post_valuation"
Hot encodings	"business_status", "ownership_status", "financing_status", "universe", "sic_codes", "naics_codes", "state", "stock_type_x", "deal_status", "deal_class", "deal_type_2"
Binary/Tertiary Conversions	"website", "paren_company", "tech_hotspot", "country", "name", "board_voting_rights"
Count Conversions	"sister_companies_count", "subsidiary_companies_count", "customers_count", "market_count", "competition", "products"
Other	"elapsed_announced_deal"-Contains number of days elapsed between the announced date and the date the deal got done and it was calculated from fields "deal_date" and "announced_date". "founder_gender"-Using gender_guesser.detector to predict the gender of the founder.

Table 1. Summary of pre-processing and transformation techniques.

2.0. The output is a numeric float. Figure 2 illustrates the distribution of valuation step-up multiples in our cleaned dataset. As shown, the majority of companies achieve less than a 2.0x multiple per round, with the 99th percentile at 10.25x.

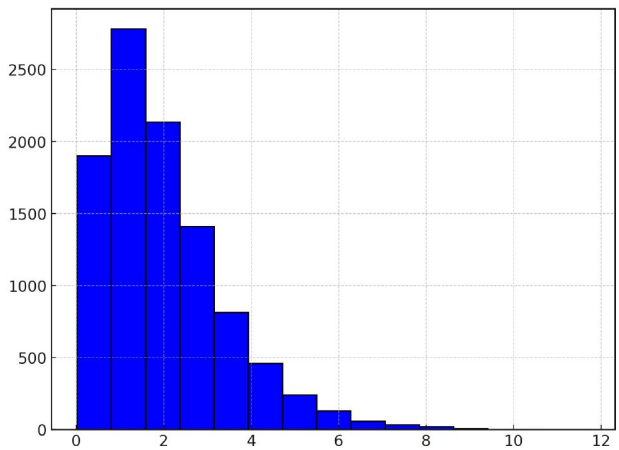

Figure 2. Distribution by fundraising round (Seed, A, B, C, D) in dataset.

3.3 Test, Development, and Training Sets

The dataset was divided into training, development, and test sets. The final split was as follows:

1. Training set: 70% of the data.
2. Development set: 10% (2,348 samples).
3. Validation set: 10% (2,347 samples).
4. Test set: 10% (2,344 samples).

4. Methods

Data manipulation and models were built using Keras [4], Tensorflow [5], Numpy [6], and Scikit-learn [7]. The architecture was designed from scratch, though inspired by previous research in this domain.

4.1 Deep Neural Network Architecture

Our final architecture was a deep fully-connected neural network consisting of a total of 10 fully-connected layers (4 hidden, 1 input, and 1 output layer). Given that our data was structured, quantitative data in tabular form, we hypothesized that a fully-connected neural network would be more suitable compared to other neural network architectures such as CNNs or RNNs.

We initialized the weights using He Normalization and applied L1 Regularization to each layer except the output layer. Between layers, the vectorized outputs were passed into a Leaky ReLU activation function, except for the output layer, where a linear activation function was used. A linear activation was chosen for the final layer as our output prediction is a real number (float type), representing the valuation step-up multiple of the company. After extensive testing and analysis (detailed in Section 5), we finalized the architecture shown in Figure 3.

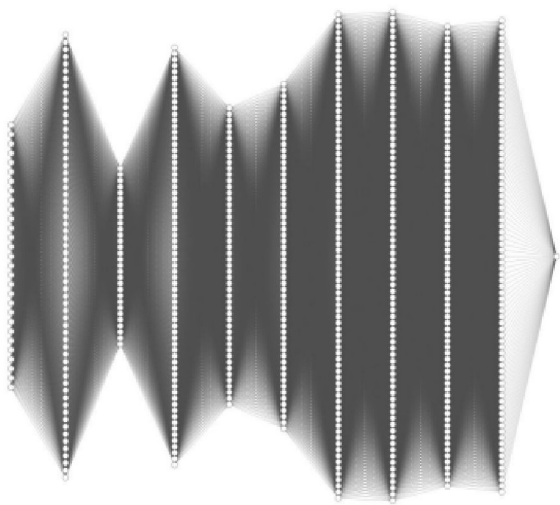

Figure 3. Deep Neural Network Architecture.

Input Layer $\in \mathbb{R}^{29}$ Hidden $\mathbb{R}^{62}$ Hidden $\mathbb{R}^{30}$ Hidden $\mathbb{R}^{70}$ Hidden R^{50} Hidden $\mathbb{R}^{28}$ Hidden $\mathbb{R}^{81}$ Hidden $\mathbb{R}^{78}$ Hidden R^{77} Hidden $\mathbb{R}^{79}$ Output $\mathbb{R}^{1}$

4.2 Hyperparameters and Loss Function

The relevant hyperparameters (outside the number of layers) were tuned to achieve the following optimal values:

- **Learning Rate:** 0.005
- **Gradient Clipping Parameter:** 0.7
- **Regularization:** L1
- **Dropout Rate:** 0.25
- **Epochs:** 100
- **Batch Size:** 64

For our loss function, we prioritized mean squared error (MSE) as our primary loss metric since the model is solving a regression problem rather than a classification one. MSE was selected over mean absolute error (MAE) due to its harsher penalties for larger errors, which are squared in MSE:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

4.3 Baseline Models

We implemented a linear regression model as a baseline for performance comparison. The linear regression model yielded an MSE of 6.6×10^{11} on the dev set, which was significantly worse than our deep learning model.

Additionally, we implemented a random forest model as an alternative baseline. We utilized the `RandomForestRegressor` class from Scikit-learn, training the model with `n_estimators = 100` and `random_state = 42`. The random forest regression model achieved an MSE of 0.97 on the dev set, providing a stronger baseline than linear regression.

5. Experiments, Results, and Discussion

5.1 Experiments

We ran our model and iterated our experiments using Google Colab, utilizing the Google TPU hardware accelerator, which significantly reduced runtime.

The learning rate was tuned manually using `keras.callbacks` a function for learning rate step decay. This function reduces the learning rate when no improvement is seen within a specified number of epochs. Adjustments to the learning rate decay and scheduling were made by monitoring the loss curve. Optimal results were achieved using an initial learning rate of 0.005, a minimum rate of 0.001, and a 0.2 factor drop per reduction.

During initial iterations, we noticed high bias, indicated by a substantial discrepancy between our training loss and that of other models, such as random forest. To combat this underfitting, we implemented gradient clipping to prevent exploding gradients, significantly reducing training loss.

Several experiments were conducted to optimize the model. The number of epochs was determined manually through trial runs, with 100 epochs chosen as the point where the training and dev losses plateaued. We also improved regularization by switching from L2 to L1 regularization, which reduced the MSE for both the training and dev sets. We hypothesize that L1 improved performance by shrinking non-relevant weights to 0, effectively acting as a form of feature selection.

	Training MSE	Dev MSE
L1	1.77	1.55
L2	1.91	2.53

Table 2. Results from comparing L1 and L2 regularization.

To optimize other hyperparameters, we used a random search process. Instead of employing the Keras Tuner library, we built a custom optimizer from scratch. We ran fifteen experiments with different seeds (0 to 14) to ensure diverse network architectures and hyperparameters, allowing effective comparisons. Seeds were used to ensure replicable experiments.

Three activation functions were tested: ReLU, Leaky ReLU, and the Swish function. The Swish function, defined as:

$$\text{Swish}(x) = x \cdot \text{sigmoid}(x) = x \cdot \frac{1}{1+e^{-x}} \tag{2}$$

is a smooth function that blends past zero and has shown promising results in academia and industry [8].

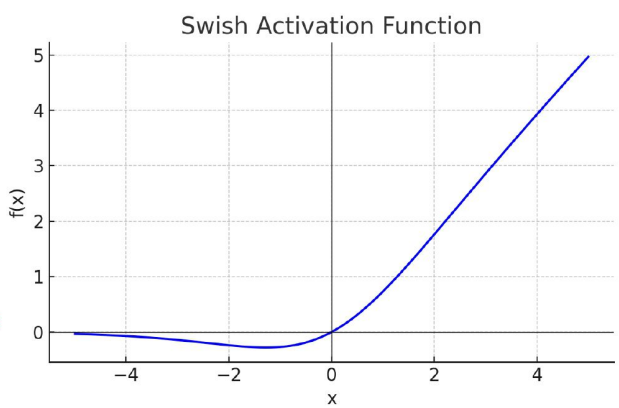

Figure 4. Comparison of activation functions: ReLU, Leaky ReLU, and Swish.

For each experiment, we calculated training, dev, and validation MSE. The dev dataset tracked loss per epoch alongside

the training set, generating plots of loss changes during training. Validation MSE was computed after training on a separate subset. Figure 3 shows a subset of hyperparameter tuning results from our random search.

	#0	#3	#4	#10	#13
Regularization	0.00637	0.00626	0.000135	0.00822	0.00775
Clipnorm	0.6	0.2	0.8	0.5	0.3
Dropout	0.05	0.2	0.3	0.05	0.05
# of layers	7	4	5	5	10
Hidden units per layer	[71, 13, 87, 25, 40, 91, 74]	[76, 4, 25, 23]	[91, 76, 5, 4, 13, 62]	[32, 93, 97, 33, 12]	[29, 62, 30, 70, 50, 58, 81, 78, 77, 79]
Activation	Leaky Relu	Leaky Relu	Leaky Relu	Relu	Leaky Relu
Training MSE	1.910	2.067	1.930	1.546	1.488
Dev MSE	1.814	1.778	1.670	1.593	1.543
Validation MSE	1.585	1.483	1.381	1.255	1.260

Table 3. Hyperparameter tuning experiments

5.2 Evaluation and Results

Table 4 compares the results of our best deep learning (DL) model with two non-deep learning baseline models.

Results		Training MSE	Validation MSE	Test MSE	Standard Deviation
Best DL Model	10-layer Leaky ReLU from hyperparameter tuning with L1	1.488	1.260	1.58	Training: 6.5 Validation: 0.61 Overall: 4.6
Linear Regression	Ordinary least squares (OLS) linear regression with L1	2.452	2.339	2.441	N/A
Random Forest	RandomForest Regressor model with n_estimators = 100	0.153	1.101	1.105	N/A

Table 4. Results of NN architecture improvements and hyperparameter tuning.

The training and dev MSE for our best model are shown in Figure 5, with error bars representing variation across multiple runs.

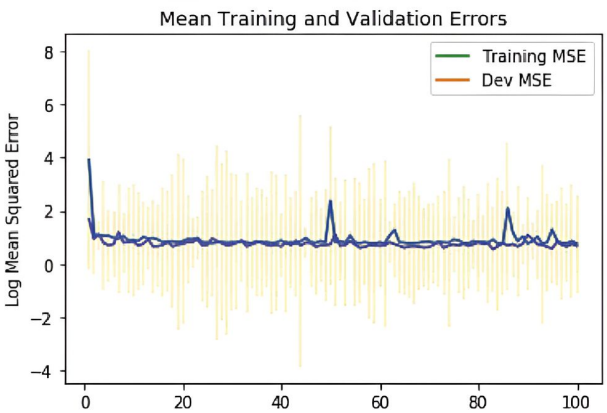

Figure 5. Mean Log MSE from 5 runs of our best DL model with error bars for Training and Dev MSE.

Our best deep neural network achieved an MSE of 1.488 on the training set, 1.260 on the validation set, and 1.58 on the test set. Across 5 runs, the standard deviation for MSE was 6.5 for training, 0.61 for validation, and 4.6 combined for training and validation.

While our deep learning model outperformed the OLS Lasso regression baseline, the random forest regressor achieved the lowest MSE among all models. This aligns with the established robustness of random forest models for structured, tabular data.

5.3 Conclusion and Future Work

Among all tested deep learning models, a 10-layer network with Leaky ReLU activation and L1 regularization performed best, achieving an MSE of 1.488 on training and 1.260 on validation.

Despite these results, the random forest model outperformed both the linear regression and deep learning approaches. Future work should explore the key differences leading to these discrepancies.

The primary limitations of this project were time and data collection. Data collection was time-intensive, and the dataset required extensive pre-processing and transformations. With additional time, we would explore implementing feed-forward neural networks, which have shown promising results for tabular data. We would also investigate alternative loss metrics and evaluation methods.

The project's source code and experiments can be found on GitHub: `https://github.com/rifaki/VC-Project`.

A. Available Columns Provided by our Dataset

Below is a comprehensive list of columns provided by the dataset:

- **Company Information:**
  - `company_id`, `company_name_x`, `familiar_name`, `previous_name`, `exchange`, `ticker`, `employee_count`, `year_founded`.
  - `business_status`, `ownership_status`, `financing_status`, `universe`, `website`, `financing_note`, `full_description`.
- **Financial Metrics:**
  - `total_raised_to_date`, `valuation_revenue`, `performance_as_of_date`.
  - `stock_price`, `average_volume`, `shares_outstanding`, `previous_close`, `price_percent_change_1_week`, `price_percent_change_4_weeks`.

  - beta, X52_week_range_low, X52_week_range_hi, market_cap_tso.

- **Location Information:**
  - location_id, location_name, address_1, address_2, city, state, zip, country.
  - location_type, location_status, office_phone, office_fax.

- **Deal Information:**
  - deal_id, deal_number, announced_date, deal_date, deal_size, pre_valuation, post_valuation, post_valuation_status.
  - deal_status, deal_class, deal_type_1, deal_type_2, deal_type_3.

- **Financial Calculations:**
  - ebitda, valuation_ebitda, valuation_revenue, valuation_cash_flow.
  - debt_ebitda, debt_equity, liquidation, revenue_percent_growth.

- **Capitalization Table:**
  - captable_id, series, stock_type_y, price_per_share_y, shares_sought, shares_acquired.
  - liquidation_preferences, dividend_rights, anti_dilution_provisions.

B. Final Features Used Within the Model:

1. **employee_count:** Number of employees.
2. **year_founded:** Year the company was founded.
3. **deal_number:** Unique deal identifier.
4. **percent_owned:** Percent acquired through the round.
5. **percent_acquired:** Percent acquired by investors in the round.
6. **pre_valuation:** Pre-money valuation in the round.
7. **raised_to_date:** Capital previously raised by the company.
8. **deal_size:** Amount raised, in millions.
9. **price_per_share:** Price per share in the round.
10. **post_valuation:** Post-money valuation in the round.
11. **business_status:** Stage of the company. Options include Clinical Trials, Product Development, Generating Revenue, Profitable, and Out of Business.
12. **ownership_status:** Who owns the company. Options include Publicly Held, Privately Held (backing), Acquired/Merged, and Out of Business.
13. **financing_status:** Current financing status.
14. **naics_codes:** Industry code dictated by the North American Industry Classification System.
15. **state:** Country of company headquarters.
16. **stock_type_x:** Type of stock. Options include Preferred, Participating Preferred, Combination, or Common.
17. **deal_status:** Status of the deal. Options include Completed and Announced.
18. **deal_class:** Type of deal. Options include Venture Capital and Individual.
19. **deal_type_2:** Type of round. Options include Series A, Series B, Series C, etc.
20. **website:** Company URL.
21. **parent_company:** Parent company of the current company.
22. **tech_hotspot:** Yes/No if the company is in a tech hotspot, sourced from Crunchbase's list of top recipient cities of venture capital funding or other identified cities.
23. **country:** Country of company headquarters.
24. **name:** Company name.
25. **board_voting_rights:** Yes/No if the board can vote.
26. **sister_companies_count:** Number of sister companies.
27. **subsidiary_companies_count:** Number of subsidiaries.
28. **customers_count:** Number of customers.
29. **market_count:** Markets the company operates within.
30. **competition:** Major competitors of the company.
31. **products:** What products the company sells.
32. **business_status:** Business status. Options include Generating Revenue, Out of Business, Startup, and Profitable.
33. **elapsed_announced_deal:** Time elapsed between the announcement date and deal date.

References

[1] Deborah Gage. The venture capital secret: 3 out of 4 start-ups fail, 2012.

[2] Alex Nakagawa et al. vc_holy_grail-1, 2017. Available at: https://github.com/Annyou/vc_holy_grail-1.

[3] Will Gornall and Ilya A. Strebulaev. A valuation model of venture capital-backed companies with multiple financing rounds, 2020. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3725240.

[4] F. Chollet et al. Keras, 2015. Available at: https://github.com/fchollet/keras.

[5] Martín Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016.

[6] T. E. Oliphant. *A Guide to NumPy*, volume 1. Trelgol Publishing, USA, 2006.

[7] F. Pedregosa, Gaël Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[8] Mary Ann Azevedo. Austin reaches top 10 in us venture markets with record funding in 2019, 2020.