CS 229 Final Project: Neural Net Classifier to Predict Unicorn Startups with Categorical Encoding of Sector Level Features

Luke Babbitt, Bradley Hu, Jenna Mansueto

Predicting

Identifying promising startups is a highly manual task for investors and venture capitalists, but by employing machine learning and building on existing research, we hope to begin to automate and support unicorn discovery. From round-level data on business financings, we built 2 business-level datasets for predicting unicorns using scikit-learn's logistic regression, support vector classification, and SGD classification. Although we encountered difficulties with the heavily imbalance nature of the dataset, we found that including more rounds of financing and upsampling the unicorn class improved model outcomes.

Data and Features

Our data comes from a Pitchbook inventory of financing deals that contains deal features (deal type, value, date), business level features (location, founders, valuation), industry features (sector, group, code), and investment features (traditional and non-traditional, lead and non-lead). Each of these provide key information at different levels of a business model that are valuable for generalizing our model to predict startups.

Datasets	Seed Round & Early Stage Series A	Total Deals
Dataset 1 (three_or_more)	at least 2	at least 3
Dataset 2 (two_or_more)	at least 1	at least 2

We transformed the data into 2 slightly different datasets with a single row for each business. These datasets had 7 business-level features, and 29 have-level features — meaning dataset 1 has 36 features in total and dataset 2 has 65 features in total.

Categorical Transformations

Embedding Encoder[1]:
scikit-learn-compliant transformer that converts categorical variables into numeric vector representations using a multi-layer perceptron; intended to encode relationships between categorical values

One-hot labels:
Creates a new binary feature for each categorical value

Embedded Industry Features

Models and Results

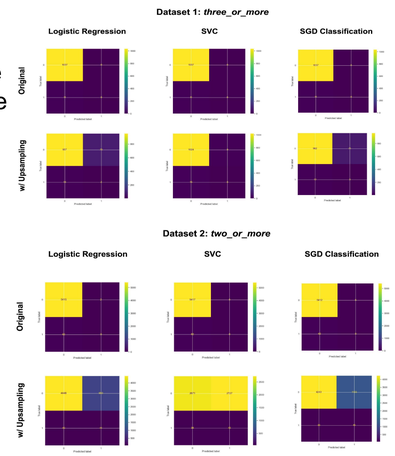
Classification Algorithms
Logistic Regression: Regularized logistic regression
SVC: Support Vector Machine for classification
SGD Classification: Implements regularized linear models with stochastic gradient descent

Upsampling Approach
Because the minority class (unicorns) represents between 0.012% and 0.015% of the two datasets, we consistently received F1 scores of 0. To combat this, we experimented with upsampling the unicorn class to balance the dataset.

	Logistic Regression	SVC	SGD Classification
Embedding	**Original Data** Dataset 1: Accuracy: 0.9895 F1 Score: 0.0 Dataset 2: Accuracy: 0.9883 F1 Score: 0.0 **Upsampled Data** Dataset 1: Accuracy: 0.9188 F1 Score: 0.0659 Dataset 2: Accuracy: 0.8110 F1 Score: 0.0750	**Original Data** Dataset 1: Accuracy: 0.9895 F1 Score: 0.0 Dataset 2: Accuracy: 0.9886 F1 Score: 0.0 **Upsampled Data** Dataset 1: Accuracy: 0.9685 F1 Score: 0.0571 Dataset 2: Accuracy: 0.6720 F1 Score: 0.0374	**Original Data** Dataset 1: Accuracy: 0.9885 F1 Score: 0.0 Dataset 2: Accuracy: 0.9883 F1 Score: 0.0 **Upsampled Data** Dataset 1: Accuracy: 0.9293 F1 Score: 0.0512 Dataset 2: Accuracy: 0.7795 F1 Score: 0.0707
One Hot	**Original Data** Dataset 1: Accuracy: 0.9895 F1 Score: 0.0 Dataset 2: Accuracy: 0.9883 F1 Score: 0.0 **Upsampled Data** Dataset 1: Accuracy: 0.9150 F1 Score: 0.0430 Dataset 2: Accuracy: 0.8193 F1 Score: 0.0764	**Original Data** Dataset 1: Accuracy: 0.9895 F1 Score: 0.0 Dataset 2: Accuracy: 0.9886 F1 Score: 0.0 **Upsampled Data** Dataset 1: Accuracy: 0.9770 F1 Score: 0.0 Dataset 2: Accuracy: 0.4953 F1 Score: 0.0301	**Original Data** Dataset 1: Accuracy: 0.9895 F1 Score: 0.0 Dataset 2: Accuracy: 0.9877 F1 Score: 0.0 **Upsampled Data** Dataset 1: Accuracy: 0.9208 F1 Score: 0.0674 Dataset 2: Accuracy: 0.7826 F1 Score: 0.0702

Initially, without upsampling, we observe very high true negative predictions, and trivial results in the remaining boxes, accounting for the observed F1 scores of 0. We observe that in both datasets, upsampling results in a slightly higher number of true positives, and a larger proportion of false positives. Accordingly, the proportion of true negatives drops.

Across both datasets, Logistic Regression results in the highest ratio of true positives to false positives.



Discussion

Overall, our biggest obstacle was the failure of our model to predict positive outcomes – F1 scores remain strikingly low across the board. This was our biggest impediment to understanding the impact of our experiments with datasets, models and categorical transformations, as this huge bias towards predicting negative outcomes rendered the effects of some of these changes difficult to interpret. That being said, while upsampling of the minority class lowered the overall accuracy of our model, it did improve the F1 scores across the board which is an improvement for the use case for our model, which is to help in identifying probable unicorns.

Classification Algorithms
As seen in the confusion matrices, logistic regression had the highest rate of true positives to false positives — an indication this matrix is more effective. However, none of the three models displayed very effective classifications.

Dataset 1 vs Dataset 2
Across all other variables, the addition of a third round of deals in Dataset 1 improves accuracy. Dataset 1 has fewer inputs but more features, suggesting the features are more significant to this issue.

Categorical Transformations
While we anticipated a performance difference between the embedding encoder and one-hot labels, our results do not show a meaningful difference in accuracy or F1 score.

Future and References

Our primary stepping stone is to try other techniques for upsampling rather than the current random upsampling implementation, which is inhibiting our model from learning. Namely, we want to look into using generative algorithms to create new unicorn rows that can comprehensively span the set of sector-level features and better parameterize the model.

We also hypothesize that lessening our criteria for what qualifies as a unicorn (e.g. by defining unicorns as companies with a valuation greater than 800 million instead of the typical 1 billion) can broaden our minority class. Training on this dataset would help maintain authentic features for model training, and could prove more beneficial than using random or generative upsampling methods which distort the underlying data distribution. We can then evaluate this model on a test set with the normal billion-dollar unicorn criteria.

References:
[1] "Embedding-Encoder." *PyPI*, pypi.org/project/embedding-encoder/.