Predicting a Startup’s Acquisition Status

Predicting

- Given a startup’s financial information, can we predict its current financial status?
- For an extremely biased dataset, a constant predictor can give high accuracy but at the cost of lower precision. How do we increase precision without sacrificing accuracy and not using over/under sampling techniques?

Data

- Kaggle dataset 'Crunchbase 2013 - Companies, Investors, etc.[1]
- Each row contains a company’s financial information and is labeled with the company’s status ('Operating', 'IPO', 'Acquired', 'Closed')
- Dataset is extremely biased:

IPO	Closed	Acquired	Operating
1.9%	3.1%	9.4%	85.6%

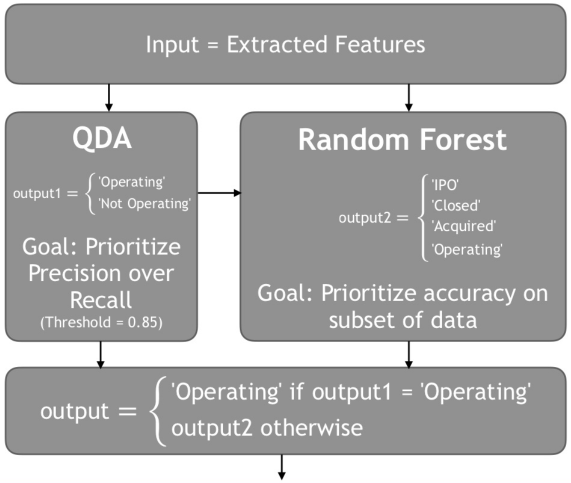
- Data is split 60/20/20 between training, validation and test sets

Features

- Dataset provides company name, permalink, category, funding dates, funding rounds, funding amount, city, state, founding dates, last milestone date
- Feature extraction: Converting qualitative data to quantitative data
  - Dates: String -> (Year, Month)
  - Locations: String -> (Longitude, Latitude, Importance) using GeoPy API[2]
- Feature selection: Forward selection to optimize features used for each model
  - Reduces overfitting while still efficiently calculated (in contrast to best subset)

Models

- Baseline:
  - input = *none*
  - output = 'Operating'
  - performs well because data is extremely biased
- Quadratic Discriminant Analysis (QDA)
- Random Forest (RF) Classifier
- Ensemble-based technique:
  - Idea: Use anomaly detection techniques to first identify a subset of the majority class with high precision. The remaining subset now has lower bias
  - Step 1: Use quadratic discriminant analysis (QDA) to first identify subset of 'Operating' classes so that the remaining data is more balanced
    - Prioritize precision by increasing threshold, since recall can be improved in Step 2
  - Step 2: Use RF classifier to classify remaining subset of data
    - Only trained on points that were not classified in Step 1 to be in majority class
  - Features used at each step are the ones obtained from feature selection for each model individually, but more specific tuning may improve performance



Results

Training Set (n = 10,636)

	Accuracy	Precision	Recall	Weighted F1
Baseline	0.85794	0.73605	0.85794	0.79233
QDA	0.85765	0.76748	0.85765	0.79277
RF	0.99953	0.99953	0.99953	0.99953
Ensemble	0.98364	0.98364	0.98364	0.98316

Validation Set (n = 3,545)

	Accuracy	Precision	Recall	Weighted F1
Baseline	0.85472	0.73055	0.85472	0.78778
QDA	0.85472	0.77849	0.85472	0.78831
RF	0.85331	0.78333	0.85331	0.79281
Ensemble	0.85614	0.79287	0.85585	0.79594

Test Set (n = 3,546)

	Accuracy	Precision	Recall	Weighted F1
Baseline	0.84659	0.71671	0.84659	0.77625
QDA	0.84602	0.71684	0.84692	0.77609
RF	0.84405	0.74416	0.84405	0.77862
Ensemble	0.84602	0.76416	0.84602	0.78005

Discussion/Future Work

Discussion

- QDA fails to increase precision because there’s not enough minority points to accurately determine the distribution
- RF increases precision, but overfits the training data, sometimes to the detriment of accuracy
- Combining a high precision model to allows us to increase precision without decreasing accuracy
- Precision increase is statistically significant, but not very large - this may be improved in future work

Future Work

- A more effective approach for this problem would be to focus on how RF can be tuned to generalize better through more effective over/under-sampling techniques
- RF models are high variance and dependent on the output of the QDA classifier. We can examine how tuning one model’s parameters and features affects the other’s
- The technique is not limited to QDA and RF. We can explore how other models can be combined using this technique