Using NLP to Predict Almost Bankruptcy

Introduction

The goal of the project is to see if sharp and extreme equity drawdown[1], as a proxy for bankruptcy, credit risk and governance, can be predicted from the language of a company's annual report and to compare this with the performance from more traditional measures using financial metrics.

While financial metrics are a generally reliable indicator of these risks, they are not perfect and often miss out on important events. The most obvious is accounting fraud or other forms of financial manipulation. Financial metrics also have a weakness when it comes to questions of general governance and over aggressive business practices. In general, they may not pick up on issues involving behavior. Exploitative labor practices, abuse of market power, misleading marketing, and concealment of product health risks are just some examples. Whether through regulatory enforcement, litigation, customer boycotts or general reputational risk, all these practices can have a severe impact on equity value and credit risk.

Alternative and unstructured data offer the opportunity to probe the predictability of such tail risk events and are of interest to a host of actors including banks, investors, rating agencies and regulators.

Model Setup

The problem is set up as a binary classification problem with the target being the maximum rolling 20-day drawdown of the equity price in the year following the release of the annual report. The target registers a positive event when the drawdown is greater than 80% and is otherwise negative. The feature set is derived from preprocessing the text of company annual reports and representing these in a tf-idf matrix. The performance of this model is then compared against a similar model using various company financial metrics as features.

The 20-day rolling 80% equity drawdown was chosen as a pragmatic indicator of company distress. An equity drawdown of this magnitude and pace threatens the solvency of any business and would be of great concern to any investor or creditor. Compared to actual bankruptcy, this "almost bankrupt" data is richer in that it is continuous and captures more events for this imbalanced dataset. It is also likely to pick out the non-financial scandals described above: sudden negative events on strong companies who are still able to survive. The 80% threshold is somewhat arbitrary but is a pragmatic choice in separating idiosyncratic drawdowns from a broad market crash. A short rolling window was chosen as sudden drawdowns are characteristic of event driven sell-offs. One calendar month (20 business days) is long enough for the market to absorb new information but too short for financials to change.

Ideally, the analysis would include other feature variables such as litigation count and regulatory actions. However, these databases are either very scant or behind expensive paywalls.

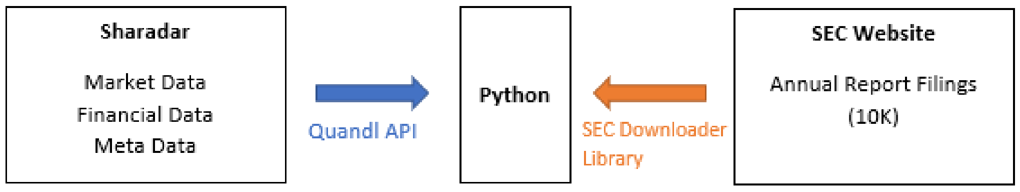
Data Bases

Sharadar offers a paid subscription covering over 20 years of history across 14,000+ US companies and is accessible through the Quandl API. Importantly for our application, the Sharadar database include bankrupt and other delisted companies. Three databases are utilized: (i) share price data, (ii) financial

[1] Maximum drawdown for the period from t_0 to t_1 is simply the maximum mark to market stock price loss over the period. More formally, it is $\left|\min\left(\frac{P_{t(i)}-\max(P_{t<t(i)})}{\max(P_{t<t(i)})}, 0\right)\right|$ for $t_0 \leq t \leq t_1$

fundamentals and (iii) metadata. Closing daily share prices are used to compute the target data drawdowns, fundamental financial data is used for the features of the comparative baseline model, and metadata has a dual purpose linking with the SEC annual report data and aiding data exploration.

Public companies are required to file annual reports (10K) with the SEC and these have been archived on the SEC website. They are electronically available through the EDGAR database and can be accessed using an existing Python package.



Data Wrangling

The data wrangling process followed eight high level steps.

Step 1: Downloading the Sharadar Data

The three Sharadar databases are downloaded through the Quandl API

Step 2: Preprocessing Target Data

The next step is to preprocess the target data as this will be used to identify those companies with positive events and ensure that we include these in the SEC annual report downloads.

The closing daily share price is used to compute the 20-day rolling drawdown across the entire Sharadar dataset since 1997. This yields a data frame with 16,973 columns representing company stock exchange tickers and 5,649 rows representing consecutive business days from Dec 1997 to June 2020. No corrections are made for missing data at this point aside from dropping rows where all entries are nan.

Step 3: Downloading the SEC Annual Reports (10Ks)

The positive event calculations are used to order a list of tickers with positive event companies appearing first. The tickers are then mapped to a CIK number through the metadata and are fed to the SEC downloader library which scrapes the company documents from the SEC website and saves it to a specified local disk.

2,450 positive event companies were identified, and the program was left to run overnight before (eventually) cancelling it the next morning. Over 200gb representing 43,536 documents from 4,370 unique CIK numbers were downloaded.

Step 4: Preprocessing Feature Data

While CIK numbers are unique, these may map to multiple tickers if companies have had name changes, acquisitions, or other changes to their stock exchange listing. The preprocessing ensures that 10Ks are mapped to all relevant tickers so that these can be matched with a company's share price across its history. Document metadata such as Filing Date is also pulled directly from the text and stored separately. A custom error log is created storing a reference to the documents that failed to process. In the event, only 13 documents failed.

Given the long period covered in the SEC database, the filed 10ks are inconsistent in format with some in text, some in html and others in XBRL. Resultantly, these documents were processed using Regex to remove special characters instead of an html parser. Further processing, such as lemmatization and stemming, were decided against on the advice of my mentor.

The end of this process yields a data frame with 44,958 rows and 15 columns. The rows represent unique 10K documents while the columns contain the ticker, filing date and processed text among other metadata.

Step 5: Merging Target and Features

Data is lost across the merge in two ways. Firstly, there are companies with market data but no 10K filings at all and those with incomplete filing databases. For example, companies with some 10Ks but not those that correspond to the year of positive events. From an initial list of 16,973 company tickers, the preprocessed 10K data base has 4,456 of these tickers. This reduces to 4,365 after the final merge providing a final accessible database of 38,807 documents. These documents are stored in a column of the final database as a string. Additional columns include tickers, Filing Date and other metadata such as current company sector. This step also includes a function that takes the maximum of the drawdown in the year following the annual report filing date and stores this target variable as a column. Missing values are removed via an inner join merge between the drawdown and 10k data frames.

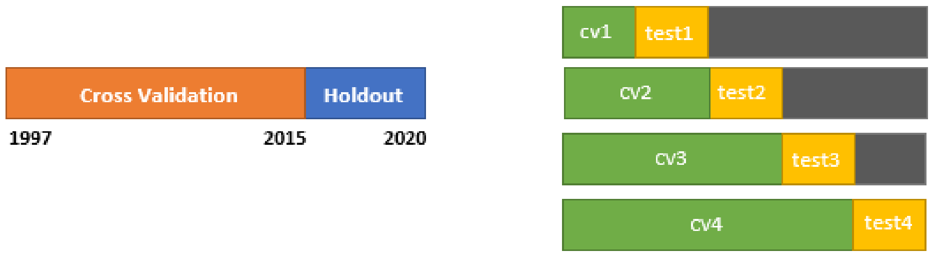
The output of this step is a final data frame with 38,807 rows being the samples and 9 columns representing the preliminary target variable, 10K text, filing date, company ticker and other useful flags or metadata for further analysis.

Step 6: Converting Features to TF-IDF Matrix

The next step is to convert the columns containing the 10K strings into a TF-IDF matrix. This is done using SK-Learn's inbuilt vectorizer function. Again, we opt for minimal processing and do not remove stop words. We add functionality that deletes vector columns containing nan. We convert the resultant sparse SciPy matrix into a spares data frame and explicitly label the columns with the vocabulary words.

Step 7: Processing Cross Validation and Holdout Sets

To expedite model validation and testing, we build various merged datasets for the cross validation and holdout sets. The holdout set consists of all data from 2015 filing dates onward and the balance is used for cross validation. The time-series cross-validation set is separated into 5 equal parts and training and testing sets are defined as below



The tf-idf vectorizer is formed on the cv training data and *transform* is then used to convert the testing documents into the model vector matrix.

Step 8: Merging Target Data and Financial Data

The purpose of the financial ratio analysis is to create a more traditional baseline against which to compare the NLP model. In this case, we use the financial data reported in the annual report and match these, via an inner join on ticker and filing date columns, with the samples obtained in *step 5*. Any rows with a nan or infinity are dropped from the resultant data frame.

Data Exploration

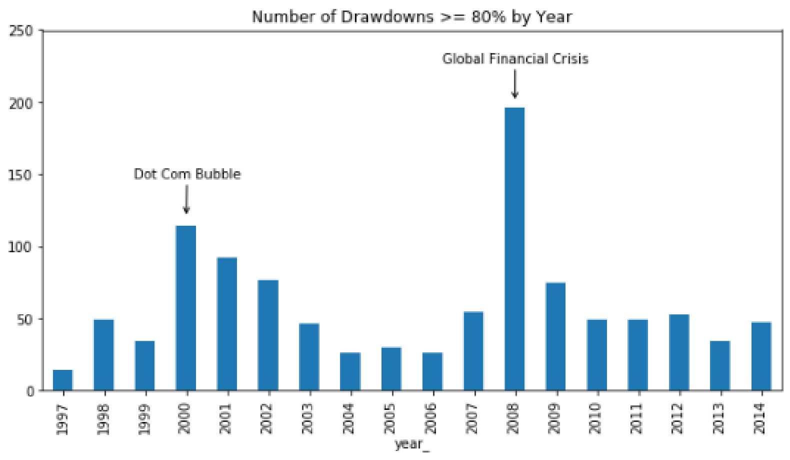
Data Imbalance

The tail risk target variable necessarily leads to a highly imbalanced dataset and the below table tracks the number of positive events and its ratio to total events across stages. The limitations on the annual report database reduces the total number of events from a potential 3,370 to only 1,414 of which 1,066 is in the model training set and 348 is in the test set. The ratio of positive events varies between 2.09% and 4.2%.

	# Samples	# Pos	Ratio Pos
Market Database	160,926	3,370	2.09%
After Matching with 10Ks	38,807	1,414	3.64%
Training Set	30,569	1,066	3.49%
Holdout Set	8,238	348	4.2%

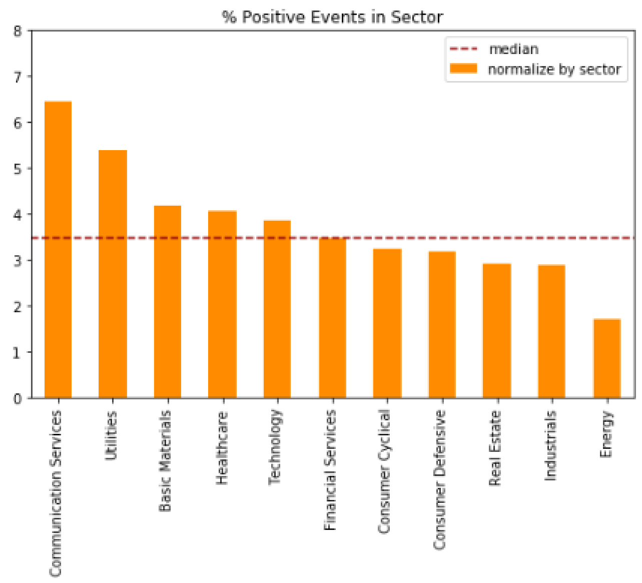
Positive Events over Time

Not surprisingly, the number of threshold drawdowns is highly variable over time with an increase over periods of US economic stress such as the 2000/01 and 2008/09 US recessions. With details are left for GitHub, we note that this does not generalize across US GDP growth. While there is evidence of a negative correlation with an expected lag (market leads GDP), the relationship is relatively weak (45% over 1998-2014) and almost entirely dependent on the extremes.



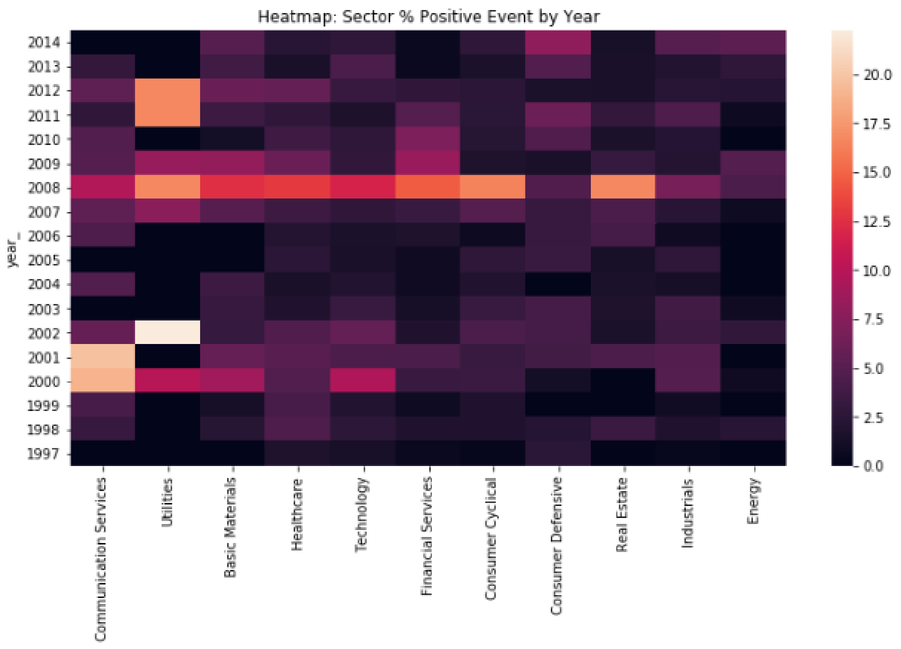
Positive Event Rate by Sector

The ratio of positive events across sectors range from 1.7% to 6.7%, with a median of 3.5% over 1997 to 2015. Surprisingly, the lowest rate is seen in energy (period excludes 2015 and 2020 oil price collapse) and the highest is in communication and services and utilities. Utilities presence near the top is surprising as they are normally associated with monopolies and stable cashflows.



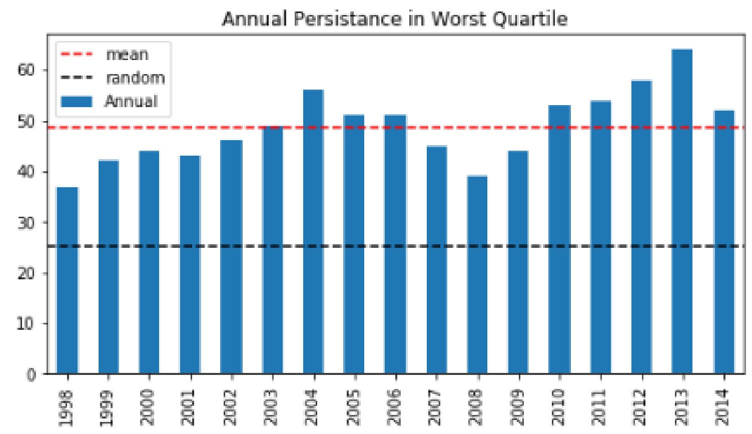
Positive Event Rate by Sector over Time

The below heatmap shows the annual sector positive event rate from 1997 to 2014. The first observation is that the 2008 GFC did not spare anyone with most sectors showing significant rates of threshold drawdown. Within that we do notice that financial services were far harder hit in 2008 than in any other year and that technology and communication services (likely to include early tech companies) were also hard hit in 2000. Again, we note that Utilities stands out in being hit hard over both recessions and again in 2011 and 2012.



Drawdown Persistence

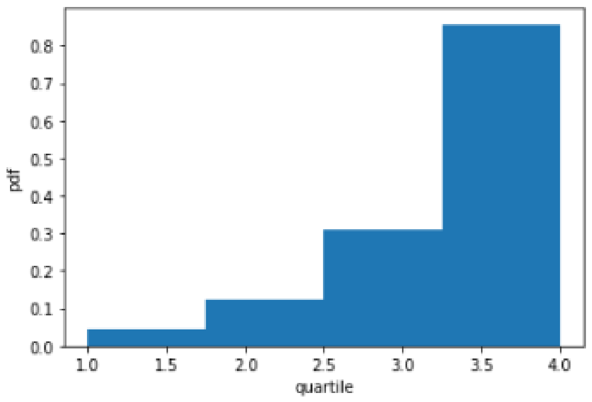
The below graph plots the percentage of companies in the current year's worst quartile who were also in the previous year’s worst quartile. Average persistence (weighted evenly by year) is just under 50% and all years are well above random chance (25%). Persistence in maximum drawdowns indicates that risky companies stay risky over long time periods. From the modeling viewpoint, it indicates autocorrelation and implies that companies in last year's bottom quartile are more likely to be in this year's bottom quartile and thus a candidate for the threshold drawdown. Interestingly, we also note that this persistence at its lowest in recession years (2001/02 and 2008/09) when the number of threshold drawdowns spike.



Drawdown Persistence of Positive Events

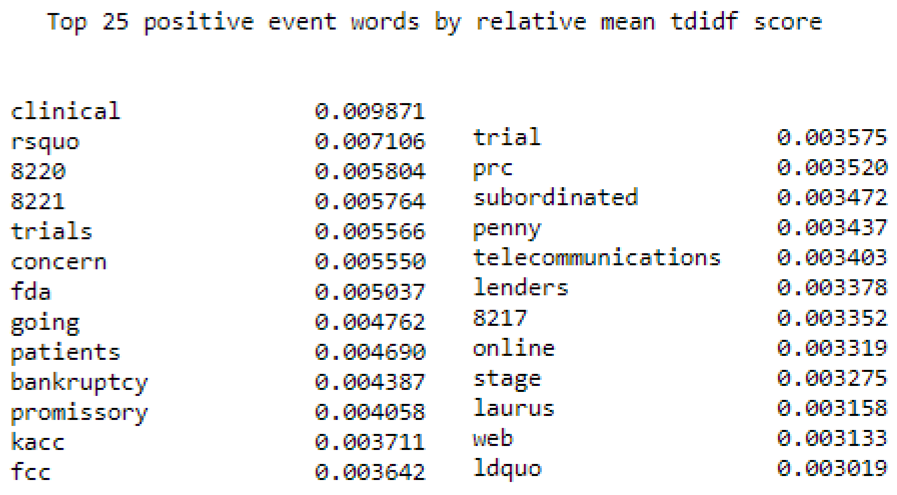
Considering the preceding analysis, the shape of the histogram is not surprising. Fully 64% of positive event companies are in the worst quartile in the year preceding the event. Just as interestingly, only 13% are in the top 50% (quartiles 1 and 2). This observation gives us some ideas of how we might use the previous year's quartile rank to increase precision and reduce false positives in the end model. Of course, it may well be the case that these are the most interesting!

```
Histogram: Previous Quartile of Company before a Positive Event

quartile  1  =  3.43 %
quartile  2  =  9.19 %
quartile  3  =  23.15 %
quartile  4  =  64.23 %
```



Highest TF-IDF Mean Words of Positive Events

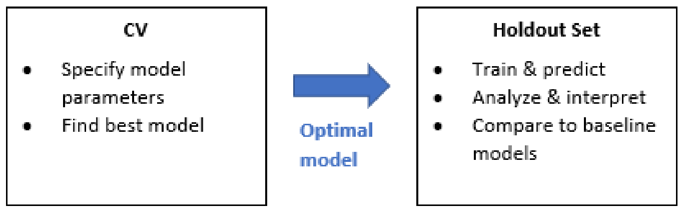
The list below shows the top 25 words of the positive event corpus relative to the entire corpus as measured by mean tf-idf score. While interpretability and predictive utility will be unclear until we start to model, it is encouraging that there are several intuitive words that center on generic debt terms such as "bankruptcy", "lenders", "subordinated" and "promissory". We can also speculate that "going" and "concern" relate to the accounting notion of "going concern" and that "trial" and “trials” relate to those of both the clinical and legal variety. Additionally, we note the presence of two federal regulatory agencies: the fda and fcc.



```
                 Top 25 positive event words by relative mean tdidf score


clinical               0.009871
rsquo                  0.007106    trial                  0.003575
8220                   0.005804    prc                    0.003520
8221                   0.005764    subordinated           0.003472
trials                 0.005566    penny                  0.003437
concern                0.005550    telecommunications     0.003403
fda                    0.005037    lenders                0.003378
going                  0.004762    8217                   0.003352
patients               0.004690    online                 0.003319
bankruptcy             0.004387    stage                  0.003275
promissory             0.004058    laurus                 0.003158
kacc                   0.003711    web                    0.003133
fcc                    0.003642    ldquo                  0.003019
```

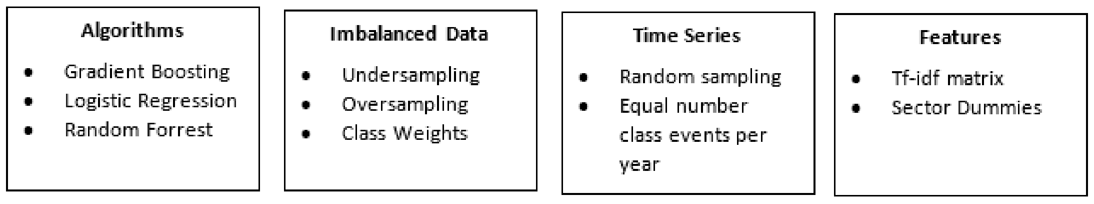
<u>Machine Learning Model</u>

The machine learning problem is structured as a binary classification model. The two nuances are that the data is (i) highly class imbalanced and (ii) in a time-series structure. We start by performing cross-validation across parameters specific to these challenges before identifying the optimal model to apply to the holdout set. This allows us to then compare the NLP model to the results of other approaches. The baseline comparison is to a similar model using traditional financial ratios and, following our observation of persistence or autocorrelation in drawdowns, we introduce another using only the previous year's drawdown quartile ranking. This latter baseline can be considered a market model as it reflects the market's assessment of short-term company risk.



Cross-Validation

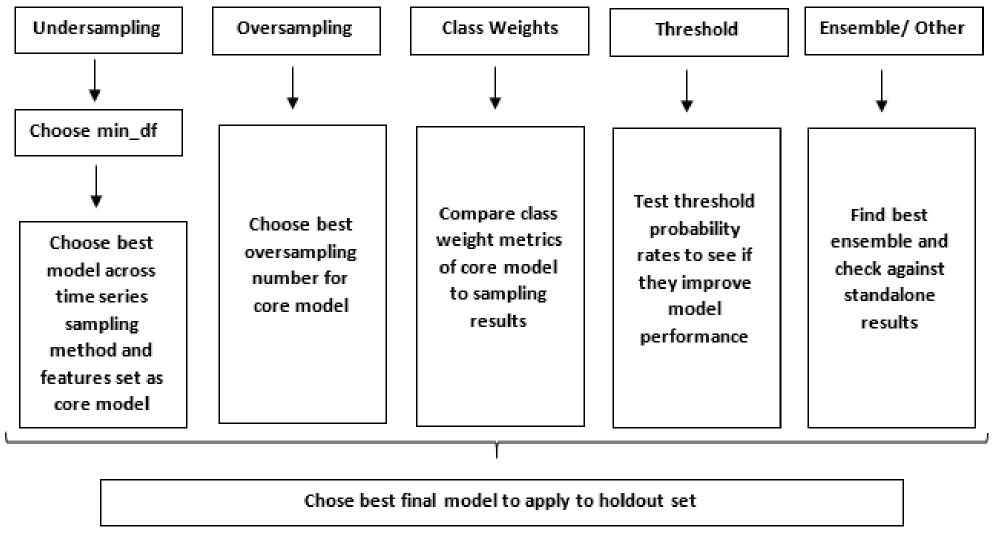
Time-series CV is performed by splitting the validation set into 5 equal parts and using an expanding window approach to define CV sets. Details can be found in the *Data Wrangling* section of the paper. We group the model parameters into 4 and use macro (harmonic) recall as the CV score:



Macro harmonic recall is simply the evenly weighted harmonic recall of the class recalls. We are more interested in recall than precision as our goal is ambitious and the analyst has many tools beyond annual report language to further scrutinize identified companies. Correctly picking out a decent sized pool of suspect companies for further analysis is more important than limiting this pool for the sake of precision. Equally, we do need a control to avoid predicting everything as a positive event! Evenly weighting class recall strikes a reasonable balance between these considerations. Finally, we note that time series CV training data, unlike regular CV, varies in amount across sets. To account for this, we weight scores accordingly. Each successive set has double the amount of data than the one preceding it and the four sets have respective weights of 10%, 20%, 30% and 40%.

Implementation Process

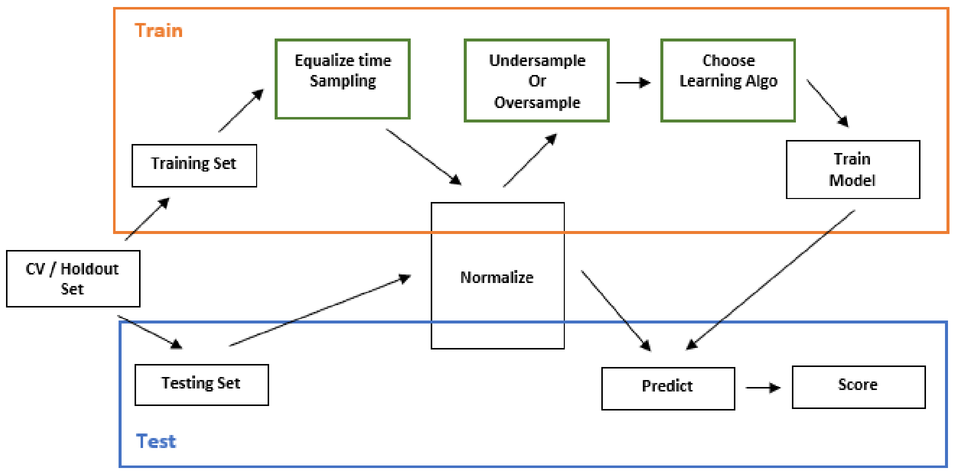
There are a lot of moving parts to consider over cross-validation and the below sketch outlines the order and process:


Undersampling
Oversampling
Class Weights
Threshold
Ensemble/ Other
Choose min_df
Choose best model across time series sampling method and features set as core model
Choose best oversampling number for core model
Compare class weight metrics of core model to sampling results
Test threshold probability rates to see if they improve model performance
Find best ensemble and check against standalone results
Chose best final model to apply to holdout set

The bulk of the work is done in *undersampling* where we choose the parameters that define the *core model* used for the rest of the project. These parameters include (i) minimum document frequency (*min_df*), (ii) random or evenly distributed sampling of classes across time, and (iii) the features set. We then run *oversampling* on the core model and take a brief look at *class weights*. The final stages (i) consider whether changing the predictive probability *threshold* away from 50% improves the previous models and (ii) looks at *ensemble* methods.

Code Implementation

The code for the learning models generally makes use of three “work-horse” functions for (i) equal time sampling, (ii) undersampling, and (iii) oversampling. The below sketches the generic process for the learning code:



Implementation Challenges

Some of the challenges or caveats that had to be deal with include:

- *Equalize time sampling:* the main challenge with the class distribution function was that it didn't make sense to simply set the number of events equal per year as this would discard most of the data for the majority class and essentially limit oversampling to drawing with replacement on both sets! Instead, the ratio of events per year of the majority class was set to be equal to the ratio of events per year of the minority class. In this way, randomly drawing from these separate samples in the same size would preserve equal distribution over time while allowing for the draw of unique data points for the majority class. The ratio was set to be 5 as this appeared practical and in line with the intended maximum oversampling number to be tested in cross-validation.

- *Normalize*: given the time series and sampling of the model, normalization had to be managed carefully to ensure (i) normalization for the test set included the training set data and that normalization for the training set was done prior to sampling so that it would include all relevant samples.
- *Sparse Matrices*: to control memory usage, the document matrices had to be kept in sparse CSR format. However, normalization required the conversion of the original dataframe to an array and there did not appear to be a way to accomplish this without creating a dense array as an interim step (20*x* more memory). This initially caused a memory leakage and crashed the program. This was solved by building a function to chunk the conversion of the dataframe into smaller pieces and then stitching these CSR matrices back together using *scipy.sparse.hstack()*

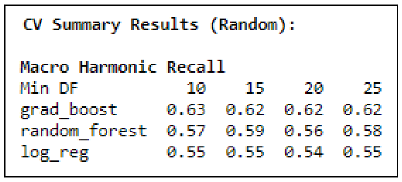
- *Undersample / Oversample*: Care had to be taken to ensure that the correct *numpy.random* functions were used in the correct place. Namely, (i) that *.sample()* was used for unique draws in undersampling and for the majority class in oversampling, and (ii) *.choice()* was used to sample with replacement for the minority class in oversampling.

- *Model training*: Interestingly, SK-Learn's learning algorithms managed to cope with mixed variable types as it simultaneously accepted X values in sparse *CSR* format and Y values in *numpy array* format.

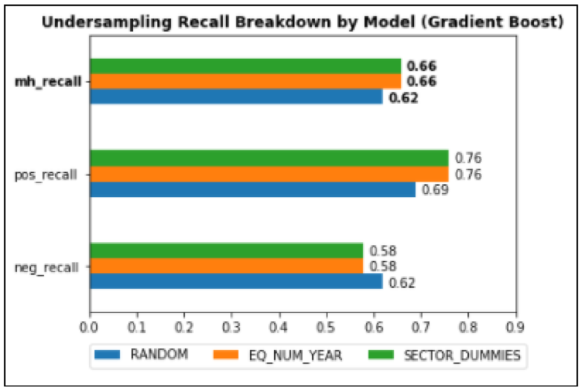
Cross-Validation Results

Undersampling and Core Model

- Undersampling's optimal score was 66%
- Gradient boost outperformed in every case
- Positive recall was favored over negative recall
- Optimal score achieved with the following parameters defining the core model
    - Min_df = 25 (little variation in performance / max computational efficiency)
    - Time equalized sampling (improves random by 4%)
    - Tf-idf matrix as features (dummy sector variables failed to yield improvement)



```
CV Summary Results (Random):

Macro Harmonic Recall
Min DF           10    15    20    25
grad_boost     0.63  0.62  0.62  0.62
random_forest  0.57  0.59  0.56  0.58
log_reg        0.55  0.55  0.54  0.55
```

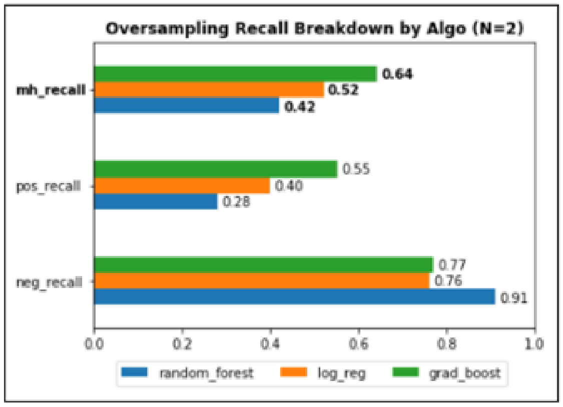


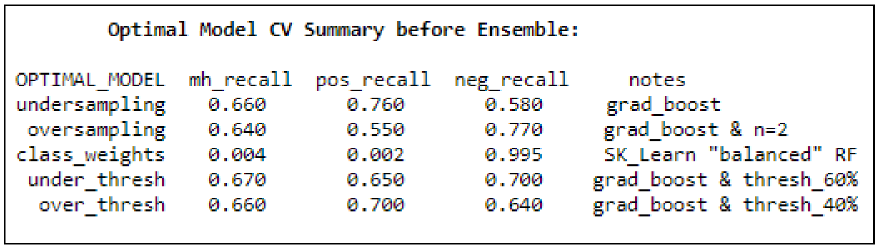
Oversampling, Class Weights and Thresholds

- Oversampling's optimal score was 64% (*oversampling number = 2*)
- Gradient boost outperformed in every case
- Oversampling favored negative recall over positive recall
- Random forest oversampling scored exceptionally low across the board as extreme overfitting to the majority class was evident

- Class weights were tested by setting SK-Learn's Random Forest model to "balanced" and the resultant score was a disappointing (!) 0.4% with only a total of *12* test samples predicted as positive (true positive events = 945)

- Undersampling optimal score improved to 67% (+1%) by increasing threshold to 60%
- Oversampling optimal score improved to 66% (+2%) by decreasing threshold to 40%



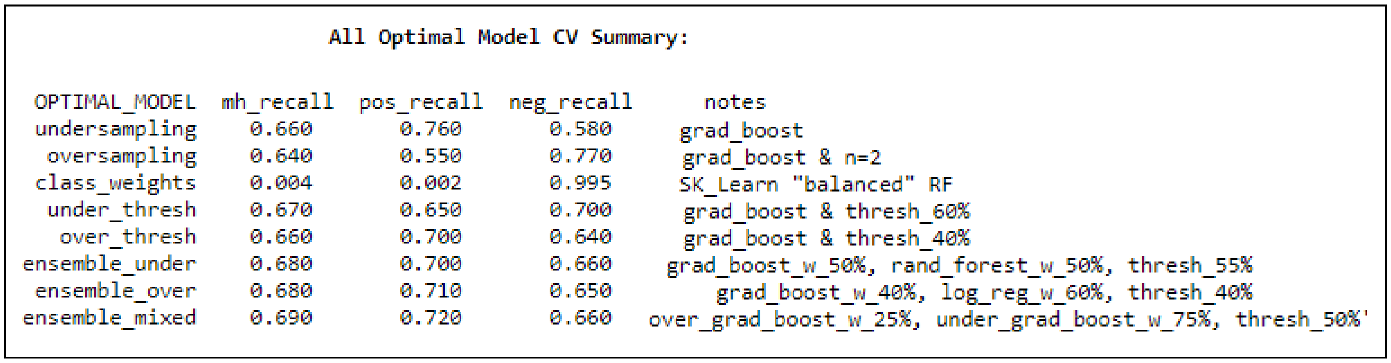


```
          Optimal Model CV Summary before Ensemble:

OPTIMAL_MODEL  mh_recall  pos_recall  neg_recall      notes
undersampling    0.660      0.760       0.580      grad_boost
 oversampling    0.640      0.550       0.770      grad_boost & n=2
class_weights    0.004      0.002       0.995      SK_Learn "balanced" RF
 under_thresh    0.670      0.650       0.700     grad_boost & thresh_60%
  over_thresh    0.660      0.700       0.640     grad_boost & thresh_40%
```

Ensemble

- Undersampling marginally improves to 68% (+1%) at a threshold of 55% and an even weighting of gradient boost and random forest
- Oversampling improves to 68% (+2%) at a threshold of 40% with gradient boosting weighted at 40% and log reg at 60%. The standard deviation of mh recall was needed as a tiebreaker for the weights

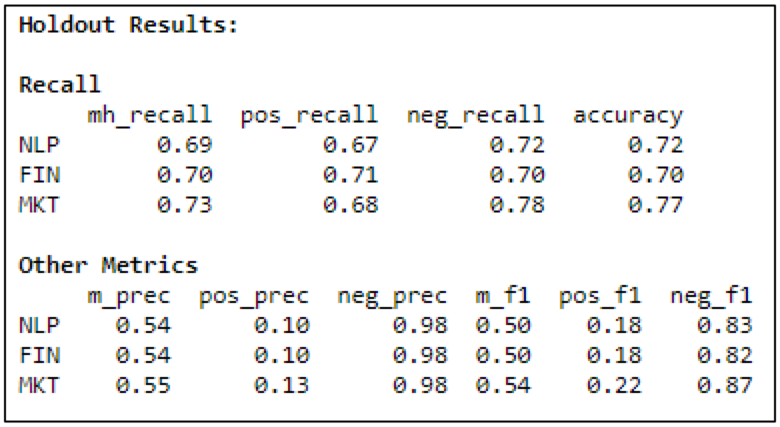
- **Mixed sampling provides the winner with optimal mh recall at 69%**
- This occurs with gradient boosting for both models, *n=3* for oversampling and a weight of 25% for oversampling and a 50% threshold.



```
                         All Optimal Model CV Summary:

 OPTIMAL_MODEL  mh_recall  pos_recall  neg_recall       notes
 undersampling    0.660       0.760       0.580      grad_boost
  oversampling    0.640       0.550       0.770      grad_boost & n=2
 class_weights    0.004       0.002       0.995      SK_Learn "balanced" RF
  under_thresh    0.670       0.650       0.700      grad_boost & thresh_60%
   over_thresh    0.660       0.700       0.640      grad_boost & thresh_40%
ensemble_under    0.680       0.700       0.660    grad_boost_w_50%, rand_forest_w_50%, thresh_55%
 ensemble_over    0.680       0.710       0.650        grad_boost_w_40%, log_reg_w_60%, thresh_40%
ensemble_mixed    0.690       0.720       0.660   over_grad_boost_w_25%, under_grad_boost_w_75%, thresh_50%'
```

Holdout Results

The mixed sampling ensemble model was used to test the hold out set with the below headline results benchmarked to the financial (FIN) and market (MKT) models. For baseline purposes, the comparison models are not optimized. Details of FIN and MKT can be found in the *Appendix*.



```
Holdout Results:

Recall
     mh_recall  pos_recall  neg_recall  accuracy
NLP       0.69        0.67        0.72      0.72
FIN       0.70        0.71        0.70      0.70
MKT       0.73        0.68        0.78      0.77

Other Metrics
     m_prec  pos_prec  neg_prec  m_f1  pos_f1  neg_f1
NLP    0.54      0.10      0.98  0.50    0.18    0.83
FIN    0.54      0.10      0.98  0.50    0.18    0.82
MKT    0.55      0.13      0.98  0.54    0.22    0.87
```

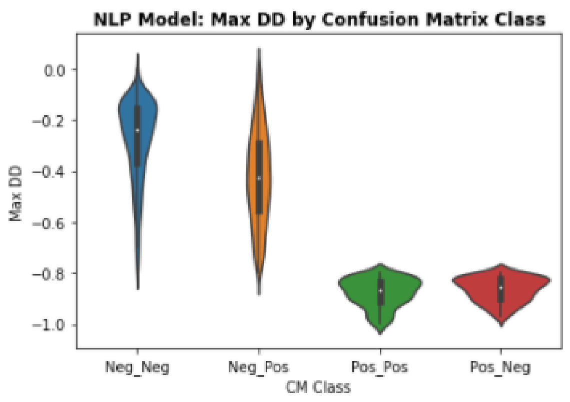
- **Given the subtlety of the information and the first-generation techniques used, the NLP model compares surprisingly well to the baseline models.** It is also encouraging that the testing results are almost identical to those of CV.

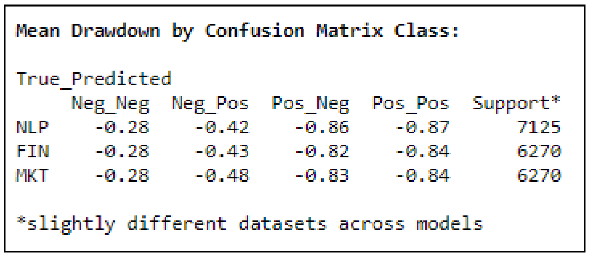
- **NLP macro harmonic recall is only 1% off the FIN model** with better negative recall but worse positive recall

- **The MKT model performs the best across all metrics** but is only 1% better than NLP on positive recall
- **Despite its simplicity, it is not surprising that the MKT model performs best as we would expect market variables to incorporate both hard financial information as well as more subtle information** such as governance, company behavior and industry pressures. The lagged nature of the feature also makes sense considering that slow changing structural flaws are likely to underpin extreme equity drawdown.
- **While a work in progress, early results suggest that combining the NLP and MKT models (using information only in the features and target set of NLP) provides a model that produces the same macro harmonic recall as MKT** but with positive and negative recall numbers transposed from the current MKT model

The Cost of Errors: Technical Perspective

One of the benefits of having a continuous variable underlying the binary target variable is that we can compute the cost of model errors by calculating the drawdown statistics for each class in the confusion matrix:



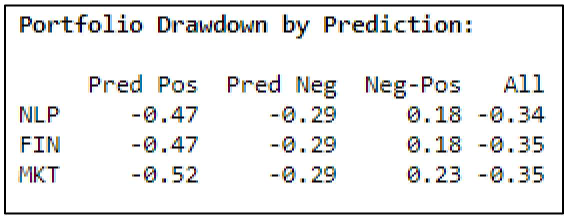


```
Mean Drawdown by Confusion Matrix Class:

True_Predicted
      Neg_Neg  Neg_Pos  Pos_Neg  Pos_Pos  Support*
NLP     -0.28    -0.42    -0.86    -0.87      7125
FIN     -0.28    -0.43    -0.82    -0.84      6270
MKT     -0.28    -0.48    -0.83    -0.84      6270

*slightly different datasets across models
```

- **All models show a similar pattern with the mean drawdown of the false positive set exhibiting far worse drawdown than the true positive set.** This is consistent across all models and suggests that the relatively high proportion of false positives and the associated low precision belie useful information about the classification space.
- While the NLP marginally lags the other models, the violin-plot illustrates the materially different statistics of the drawdown across confusion matrix classes

The Cost of Errors: Business Perspective

From a business viewpoint, the predicted positive companies would be *valued* lower than the other companies. For an investor (equity or credit), this means one could short (or not invest) in the predicted positive companies and go long the rest. A good metric to get a sense for the return on the strategy would be to look at the difference between the evenly weighted predicted positive portfolio versus (i) a similar portfolio of the rest and (ii) an evenly weighted portfolio of all companies. To be clear, these are not the actual returns (different to drawdowns and not considering time mismatches between max drawdowns) but still gives a good sense of portfolio performance in metrics consistent with the risk-oriented model target:



```
Portfolio Drawdown by Prediction:

     Pred Pos  Pred Neg  Neg-Pos   All
NLP     -0.47     -0.29     0.18 -0.34
FIN     -0.47     -0.29     0.18 -0.35
MKT     -0.52     -0.29     0.23 -0.35
```

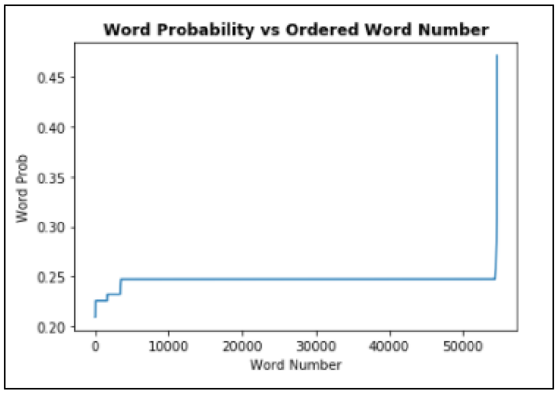
- **While the MKT portfolio performs slightly better, NLP still demonstrates strong "returns"**

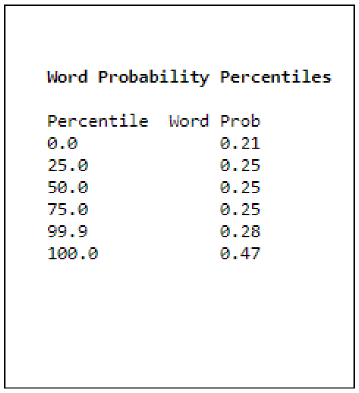
Words Interpretation

Interpretation of the model is somewhat challenging, but we can look at the individual words that have the highest probability associated with a positive event. In this section, we limit the analysis of the words to those found only in the holdout set defined by the years 2015 to 2019. This is done to get a better idea of the words explaining the holdout results and not words that might be important based on past events but with no explanatory value on the unseen documents. The caveat throughout this section is that a word can have a high probability while only appearing in one or very few documents that happen to have a positive event. In that respect, not all words will be *generally* predictive.

Word Probabilities

The word probabilities are generated by forming a document matrix where each document only has one unique word and then running this through the model prediction for each annual holdout set. For repeated words, the maximum probability is chosen.





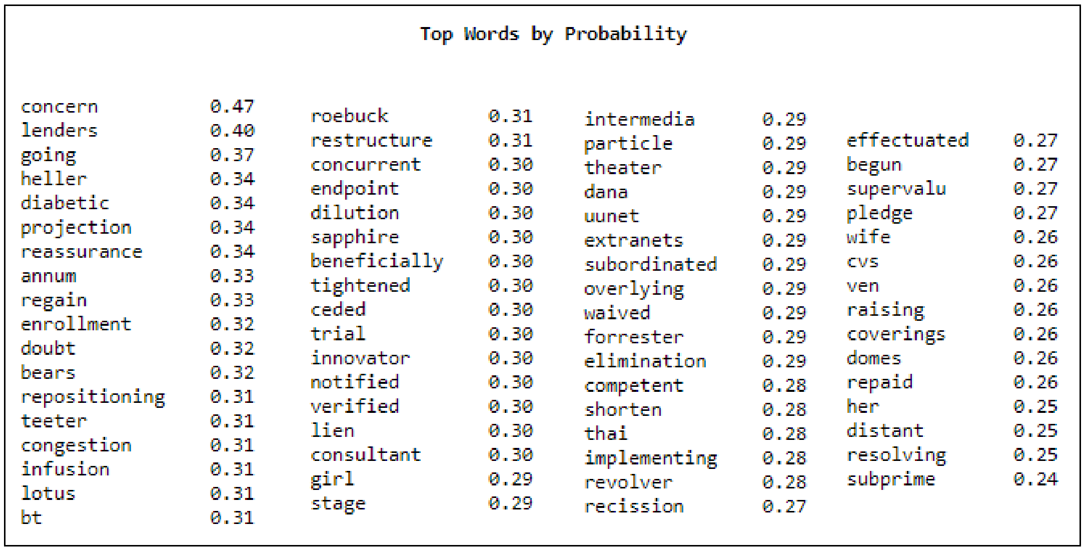
```
Word Probability Percentiles

Percentile  Word Prob
0.0              0.21
25.0             0.25
50.0             0.25
75.0             0.25
99.9             0.28
100.0            0.47
```

- **The maximum single word probability is below 50% and 99.9% of the words are between 20 and 30%. This indicates that it is the combination of words found in the document and not single implicating words that produce documents of high probability.** Given the nuance of the underlying task and the caveat above, this is reassuring.

Top Words

The words below represent the words with the highest probabilities across the holdout sets. They were generated by aggregating the top 20 words from each holdout set. Repeated words take the maximum probability found across the holdout years.

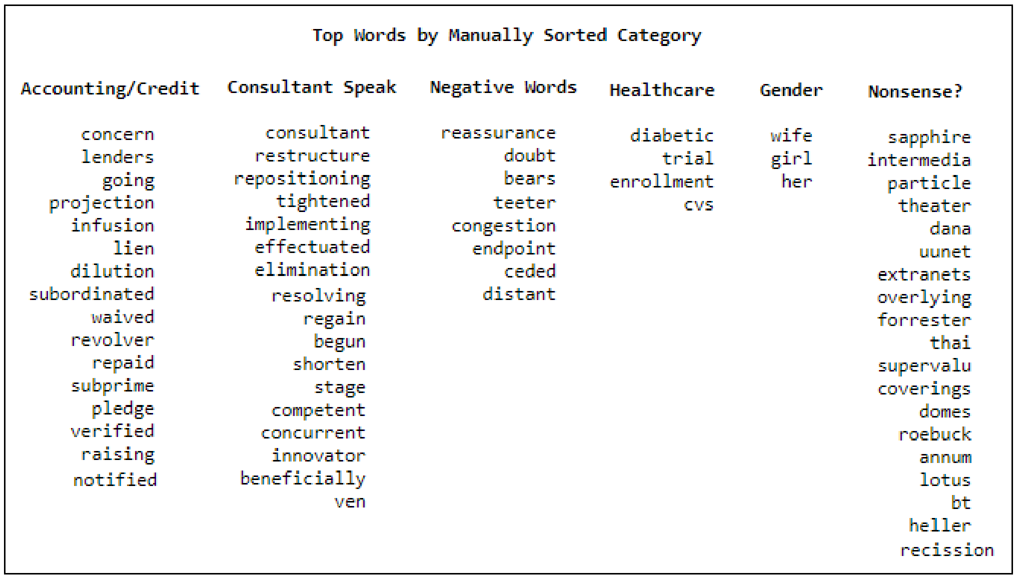
The table below lists these top words in descending order of probability:



Top Words by Probability

concern	0.47	roebuck	0.31	intermedia	0.29	effectuated	0.27
lenders	0.40	restructure	0.31	particle	0.29	begun	0.27
going	0.37	concurrent	0.30	theater	0.29	supervalu	0.27
heller	0.34	endpoint	0.30	dana	0.29	pledge	0.27
diabetic	0.34	dilution	0.30	uunet	0.29	wife	0.26
projection	0.34	sapphire	0.30	extranets	0.29	cvs	0.26
reassurance	0.34	beneficially	0.30	subordinated	0.29	ven	0.26
annum	0.33	tightened	0.30	overlying	0.29	raising	0.26
regain	0.33	ceded	0.30	waived	0.29	coverings	0.26
enrollment	0.32	trial	0.30	forrester	0.29	domes	0.26
doubt	0.32	innovator	0.30	elimination	0.29	repaid	0.26
bears	0.32	notified	0.30	competent	0.28	her	0.25
repositioning	0.31	verified	0.30	shorten	0.28	distant	0.25
teeter	0.31	lien	0.30	thai	0.28	resolving	0.25
congestion	0.31	consultant	0.30	implementing	0.28	subprime	0.24
infusion	0.31	girl	0.29	revolver	0.28		
lotus	0.31	stage	0.29	recission	0.27		
bt	0.31						

To interpret the words, we group them into manually chosen categories:



Top Words by Manually Sorted Category

Accounting/Credit	Consultant Speak	Negative Words	Healthcare	Gender	Nonsense?
concern	consultant	reassurance	diabetic	wife	sapphire
lenders	restructure	doubt	trial	girl	intermedia
going	repositioning	bears	enrollment	her	particle
projection	tightened	teeter	cvs		theater
infusion	implementing	congestion			dana
lien	effectuated	endpoint			uunet
dilution	elimination	ceded			extranets
subordinated	resolving	distant			overlying
waived	regain				forrester
revolver	begun				thai
repaid	shorten				supervalu
subprime	stage				coverings
pledge	competent				domes
verified	concurrent				roebuck
raising	innovator				annum
notified	beneficially				lotus
	ven				bt
					heller
					recission

While the list of very specific and likely spurious words is relatively lengthy, the rest appear to form three natural and unsurprising groups:

- **Accounting / Credit**
- **Consultant Speak / Business Reorganization**
- **Negative sentiment words**

Healthcare words were highly appearing in *Data Exploration*, but these industry words are relatively few in the holdout set after learning. Gender is included as it is both amusing and, potentially, insightful. While we do not know the context nor general applicability of these words (likely low), it would be apt to find such inappropriate words in the statements of badly governed companies.

Future Work

- It would be interesting to see whether ensemble, stacking or joint features engineering across the three models results in better predictions.
- Running the model with second generation NLP tools (word2vec seq2seq) better able to parse meaning could considerably improve performance with different interpretive potential. For example, the fuzzy logic of *word2vec* algebra could lead to some interesting insights.
- Preprocessing the text could be improved by (i) taking care of the tagging / formatting codes still evident in the word matrix and (ii) removing proper nouns. Comparing current results with the removal of proper nouns would be particularly interesting as speculative arguments could be made for both sides. My own intuition falls on the side of keeping the names as it seems plausible that proper nouns could have predictive capability for our application: common dodgy jurisdictions, questionable auditors, high risk lenders etc.
- Running the model on a features set of weighted moving average tf-idf vectors per company could prove fruitful. Following on from the observation of autocorrelation in the market model, we might expect past documents (cumulative information) to be more useful than a 1-year snapshot.
- Running the model on the change in annual (or annual moving average) company tf-idf as the features set could amp performance. Abrupt changes in key words or *n-grams* could indicate high short-term risk. It would also be interesting to run this change together with the "stock" tf-idf matrix. The intuition being that stock variable tells us which companies are generally risky and the delta indicates higher chances of this being reflected in a nearby positive event.
- Interpretation could be aided by running an LDA for the features set and seeing which categories are the most predictive. We already see hints in the results of the top words in the current model and being able to identify categories for leverage, environmental considerations, and gender attitudes could be fascinating with applications in the hot area of ESG.
- Running the model as a multilabel classification model on the drawdown data. Autocorrelation might be hard to beat but, following from the analysis of drawdowns by confusion matrix class, it would eb interesting to see how NLP prediction carves this classification space.
- Calculating actual performance of a portfolio of evenly weighting short predicted positive versus long predicted negative as the CV score. Again, this follows from the analysis of the holdout results.

Appendix

A1: Baseline FIN Model

The FIN model takes annual financial information from the annual balance sheet *(Sharadar fundamental database*) and uses this to compute various financial ratios consistent with solvency models. Calculations are found in the codebase on GitHub.

The model uses the following common financial ratios as the feature set:

- Net Income / Assets (return on assets)
- Assets / Liabilities (leverage / solvency)
- Net Cash from Operations / Net Income (accruals)
- Net Cash from Operations / Liabilities (debt servicing)
- Net Cash from Operations / Equity (core operational return on equity)
- Net Income / Equity (return on equity)

Gradient boost oversampling was used with the number per sample equal to 3 time the positive set size. Just as in the NLP case, the samples were distributed evenly over time. The model was not optimized and is not meant to represent the best financial features model out there. After all, the NLP model only uses first generation tools and does not represent the most state of the art NLP model. Rather, the idea is to see if the basic NLP model can compete with a basic financial ratio model.

A2: Baseline MKT Model

Autocorrelation (or momentum) is a common feature of market prices and it is no surprise that we see this evident in the drawdown data (see *Data Exploration*). It is also intuitive that risky companies, whether related to management style or inherent in the industry, remain so for extended periods. Absolute market prices can vary for a number of macroeconomic reasons and it is more traditional in quantitative finance to take relative rankings. Quartiles are a familiar ranking system and also happen to be convenient for the classification problem.

The model uses the following as the features set:

- 1-year lagged drawdown quartile of the company

It is also trained with gradient boost using oversampling (n = 3) and with the same time-series adjustment to the samples as used previously.