

**ISSS609 Text Analytics and Applications**

**Project Report**

**Group 5 Members:**

Catherine Leo

Desmond Ho

Kevin Moe

Lim Hui Ting

Ong Hui Lin

Siddhant Agrawal

6 Aug 2019

**Introduction**

1. **Background**. Investment analysts at large buy-side financial institutions receive about 2,000 research reports daily from research institutions and other banks. The content of these reports can range from something as broad as an economy outlook to something much narrower like a company outlook. Typically, most reports have a clear “Analyst View” attached to it indicating if the analyst wishes to Buy, Sell or Hold the stated security.
2. **Problem**. While analysts today have access to this enormous amount of research reports, they are unable to use it to make investment decisions as the information is not segregated and structured. Manually sorting and analyzing thousands of reports daily is time consuming, if not impossible. This could result in investment recommendations being made based on an incomplete analysis of information available.
3. **Proposed Solution**. We propose to develop a solution that will **(1)** extract text data from reports, **(2)** Cluster research reports into their respective industry sectors, **(3)** carry out sentiment analysis to determine the investment sentiment[[1]](#footnote-2) of each report and **(4)** use Named Entity Recognition to extract names of companies from optimistic reports. Ultimately, this will enable the analysts to rapidly access a large volume of research reports and extract the information of interest, leading to considerable time savings.
4. **Ideation.** As part of our ideation process, the team sought inputs from a staff of an Equities department of a local bank who faced this problem, and learned that a they were seeking to establish a similar in-house recommendation platform. Our project is modelled after their requirements.

**Solution Overview**

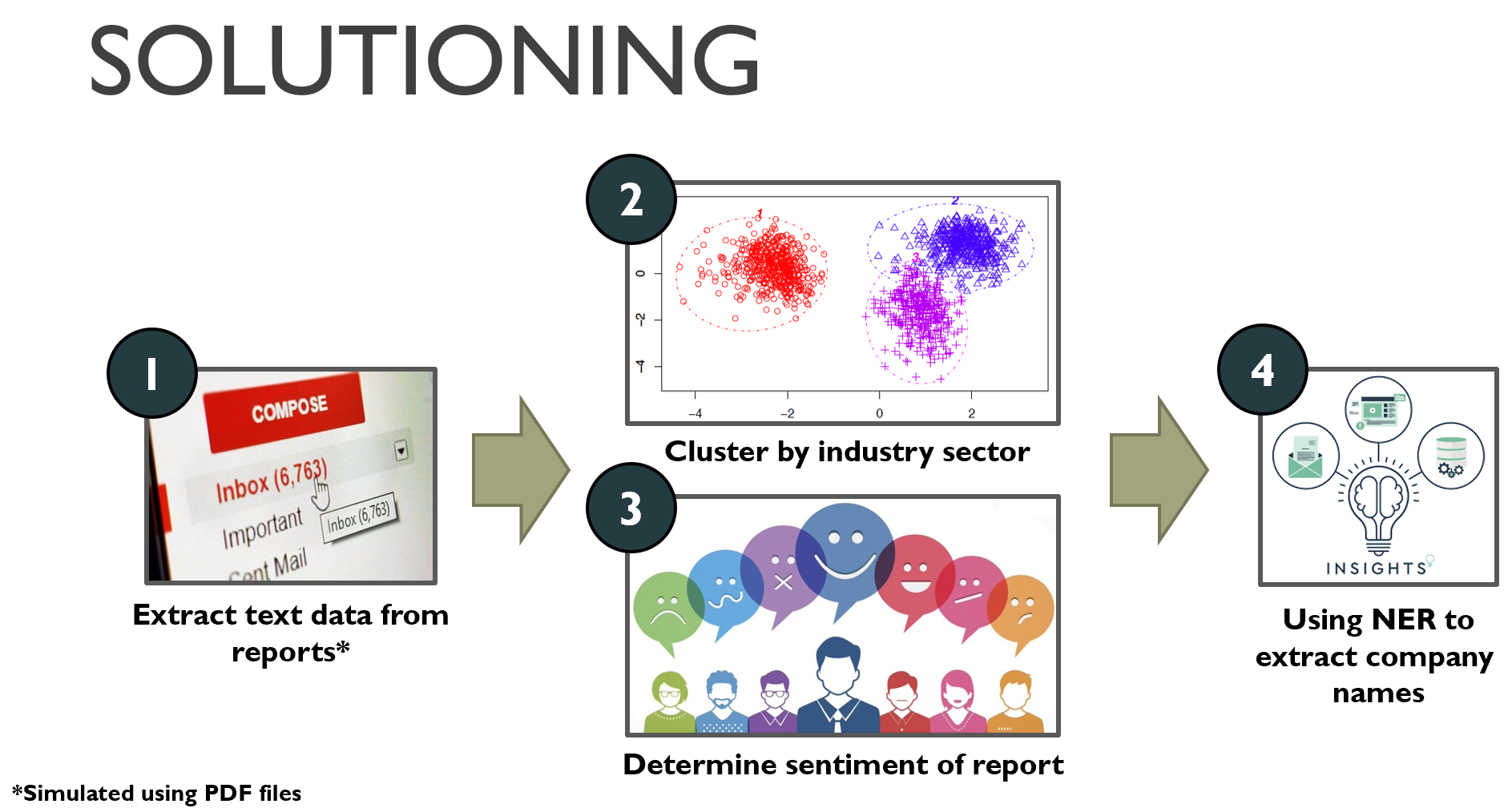


Figure 1. Overview of proposed solution.

1. We propose a 4-step approach for the solutioning.
   1. **Extraction**. Text data were first extracted from the investment research reports which were in PDF format. In reality, the reports could be in the form of emails, but we opted to simulate the problem using PDFs as they were easier for us to access.
   2. **Clustering**. The text data were subsequently clustered into their respective industry sectors. We explored three different approaches to feature engineering before carrying out k-means clustering.
   3. **Sentiment Analysis**. We carried out sentiment analysis to determine the investment sentiment of each report. Two sentiment categories were used – **(1)** optimistic and **(2)** not optimistic.
   4. **Named Entity Recognition (NER)**. Lastly, using NER, we extracted the company names from the reports that were deemed optimistic. Doing so would allow analysts to have a quick oversight of promising companies. This was done as a demonstration to complete the functionality of our proposed solution. As such, we will not be going into the details of this implementation.

**Solution Details**

1. **Extract Text Data from Reports.** Text data were extracted from the PDF file using PyPDF2 from python library. These text files were subsequently loaded into python and tokenised using NLTK package. Text preprocessing such as Lower(), Lemmmatization() and stopwords removal were also performed on the corpus. One of the challenges was that the extracted text data from PDFs was not clean. Most of the noisy data was from captions associated with charts, graphs and report appendices. Fortunately, the PDFs were quite consistent in their formatting, allowing a rule-based approach to identifying the sections that were relevant for scraping.
2. **Clustering by Industry Sector.** Clustering was performed using the k-means algorithm from Scikit-learn. The number of clusters was a known parameter as the bank would have subscribed for the investment reports. 4 industry clusters were chosen: **(1)** Banking, **(2)** Energy, **(3)** Real Estate and **(4)** Healthcare. Three approaches of extracting features were explored.
   1. **Approach 1 -** The first approach was to identify the top ***k*** common words in the corpus (***k*** is a parameter ranging from 200-600). These ***k*** words would be used to create feature vectors using term frequency for the k-means algorithm (see Fig.2).
   2. **Approach 2 -** The second approach was an extension of the first. We selected the top ***k*** common words, and further used the word document frequency (***DF***) as a parameter. Intuitively, words that occurred in too many documents would be less discriminative (high ***DF***). Thus, the maximum ***DF*** was used to control which words would be used for the clustering based on its distribution within the corpus (***DF*** ranged from 0.1-0.9). In all, words that satisfied both the ***DF*** and ***k*** values were used for TFIDF vectorization (see Fig.2).

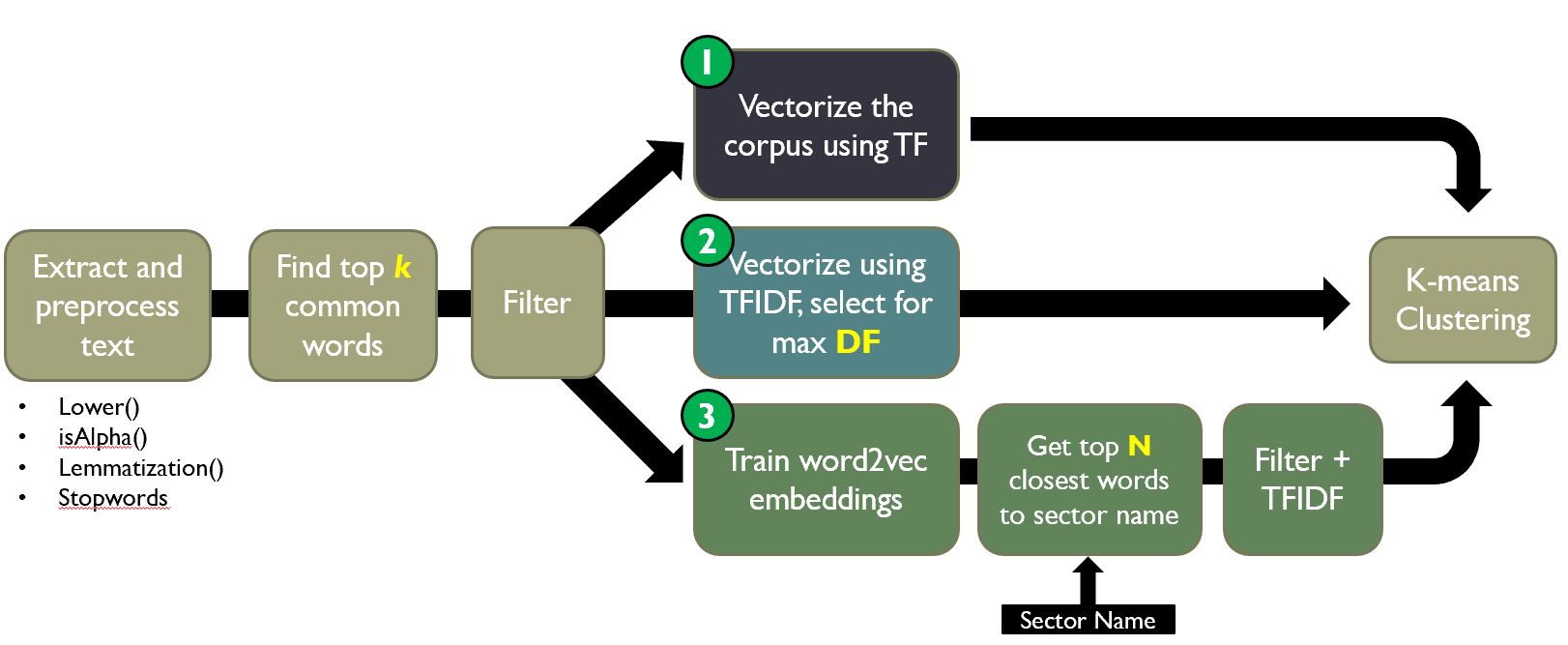


Figure 2. Three approaches to clustering.

* 1. **Approach 3** – Here, first selected for the top ***k*** words, and then trained word2vec embeddings using the Gensim package using a skip-gram[[2]](#footnote-3) configuration. Thereafter, we used the respective industry sector names to extract the ***N*** closest words for each sector. Finally, TFIDF was then used to carry out the clustering (see Fig.2).
  2. Since the cluster categories were fairly balanced (4 industry sectors with ~100 documents each), we used purity as a metric to determine the efficacy of our clustering.

1. **Sentiment Analysis.** In the financial markets and investment community, people usually speak about investment or market sentiment. This sentiment is usually referred to as the amalgamation of participant’s attitudes in the investment world. They are usually referred to as ‘bullish’ (or optimistic), wherein participants feel that the price of an asset will rise, and ‘bearish’ (or non-optimistic) on the other hand, wherein participants feel that the price of an asset will decline.
   1. **Acquiring sentiment labels.** In order to classify the 400 equities investment reports at our disposal, our team took time to determine the sentiments of the reports and classify them according to whether they were positive or negative. We manually browsed through 400 reports by dividing the work up between the 5 of us, to label the reports as positive or negative.
   2. **Developing a sentiment lexicon.** The sentiment lexicon was created using common jargon and terms used by the industry to describe the outlook of a security. Since 4 team members had background experience in the finance sector, we used our domain knowledge to develop a sentiment lexicon. See Fig.3 for words used in our lexicon.
   3. **Filtering the corpus**. The corpus was then filtered to select for only the tokens present in the lexicon i.e. terms which were outside the lexicon were removed. After this, the reports were converted to TFIDF vectors using Sklearn’s CountVectorizer function.

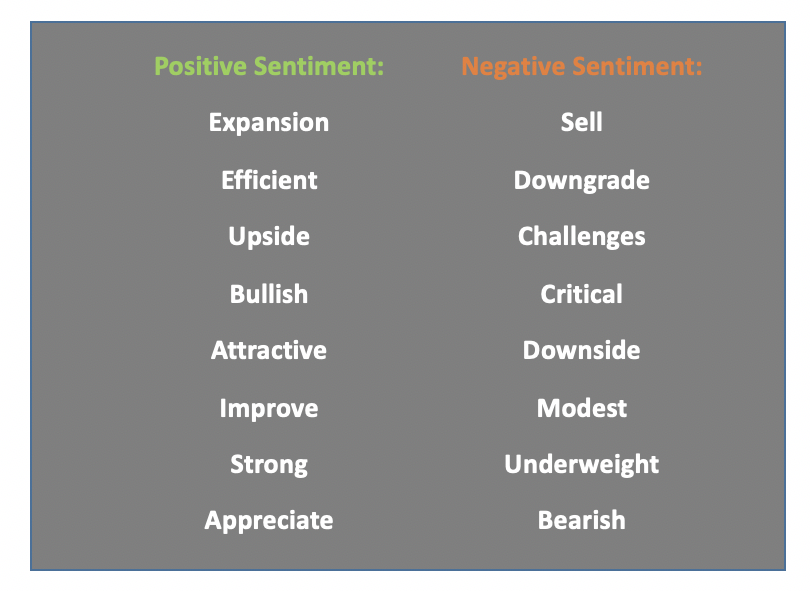


Figure 3. Example of sentiment words associated with financial reports.

* 1. **Classification.** We discovered that there was indeed a class imbalance: The majority class was had optimistic sentiment (70%), while the minority class had non-optimistic sentiment (30%). This indeed reflected the inherent bias present in the industry for publishing more reports with positive recommendations. In order to overcome the class imbalance, we conducted random oversampling on the minority dataset, whereby the training data was supplemented by multiple copies of some of the minority classes. Other parameters used for the classification are provided in the table below.

Table 1. Summary of Logistic Regression Parameters

|  |  |
| --- | --- |
| **Parameters** | **Remarks** |
| 70% Train – 30% Test split | As logistic regression was a supervised model, we had to split the dataset into training and testing instances. We did not employ a validation set as we did not further compare this model with other classifiers. Note that the training dataset was class-balanced to contain equal numbers of optimistic and non-optimistic reports. However, balancing of the test set was not necessary. |
| L2 Regularization | L2 Regularization was carried out to avoid overfitting. This effectively adds a term to the regression cost function that ensures that less relevant lexicon words were given weights that were close to zero. |
| Metrics | We present all metrics as part of this report. However, the business case required us to highlight promising reports. As such, the set of reports that were classified as optimistic had to be as ‘pure’ as possible i.e. high precision. |

**Results and Analysis**

|  |  |
| --- | --- |
|  |  |

1. **Clustering results**. The table below summarizes the result from the three approaches to clustering.

Table 2. Summary of Clustering Results. Best results were achieved by Approach 2.

|  |  |  |  |
| --- | --- | --- | --- |
| **Approach** | **Features** | **Hyper Parameters** | **Best Results (Purity %)** |
| 1 | Top **k** words (TF) | **k** | 47 |
| 2 | Top **k** words (TFIDF) | **k**, max document frequency (**DF**) | 97 |
| 3 | Closest **N** word vectors | **k**, **N** | 96 |

* 1. **Approach 1**. Clustering using the TF approach had a relatively lower purity score than the other two approaches. The purity score was about 47% at ***k*** common words varying between 200-600. To visualize the clustering, T-SNE[[3]](#footnote-4) was used to reduce the vector dimension from k to 2 for visualization purpose. From the T-SNE plot, only 2 clusters could be observed. However, in fact, there were 4 industry sectors present in the corpus. This partly explained why the purity results were poor.

|  |  |
| --- | --- |
|  |  |

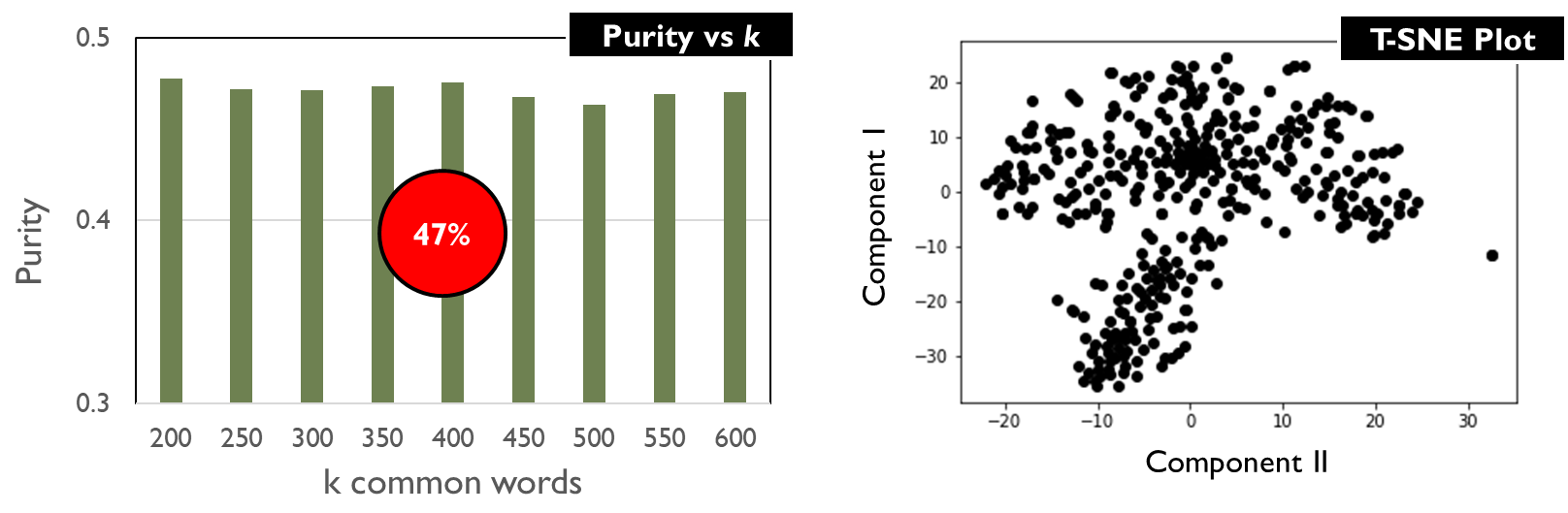


Figure 4. Purity vs k (left); T-SNE plot (right) showing an ambiguous number of clusters.

* 1. **Approach 2**. Using TFIDF instead of TF, the results for this approach were much more promising. Fig. X below shows how purity changes with both parameters **k** and **DF**. Based on the heat map, the highest purity score was about 97% at ***k***=200 for common words and max**DF**=0.4. We note that purity did not change when maximum **DF** was increased beyond 0.4, possibly because the top **k** words were not present in more than 40% of the documents. The corresponding T-SNE plot also showed 4 distinctive clusters.

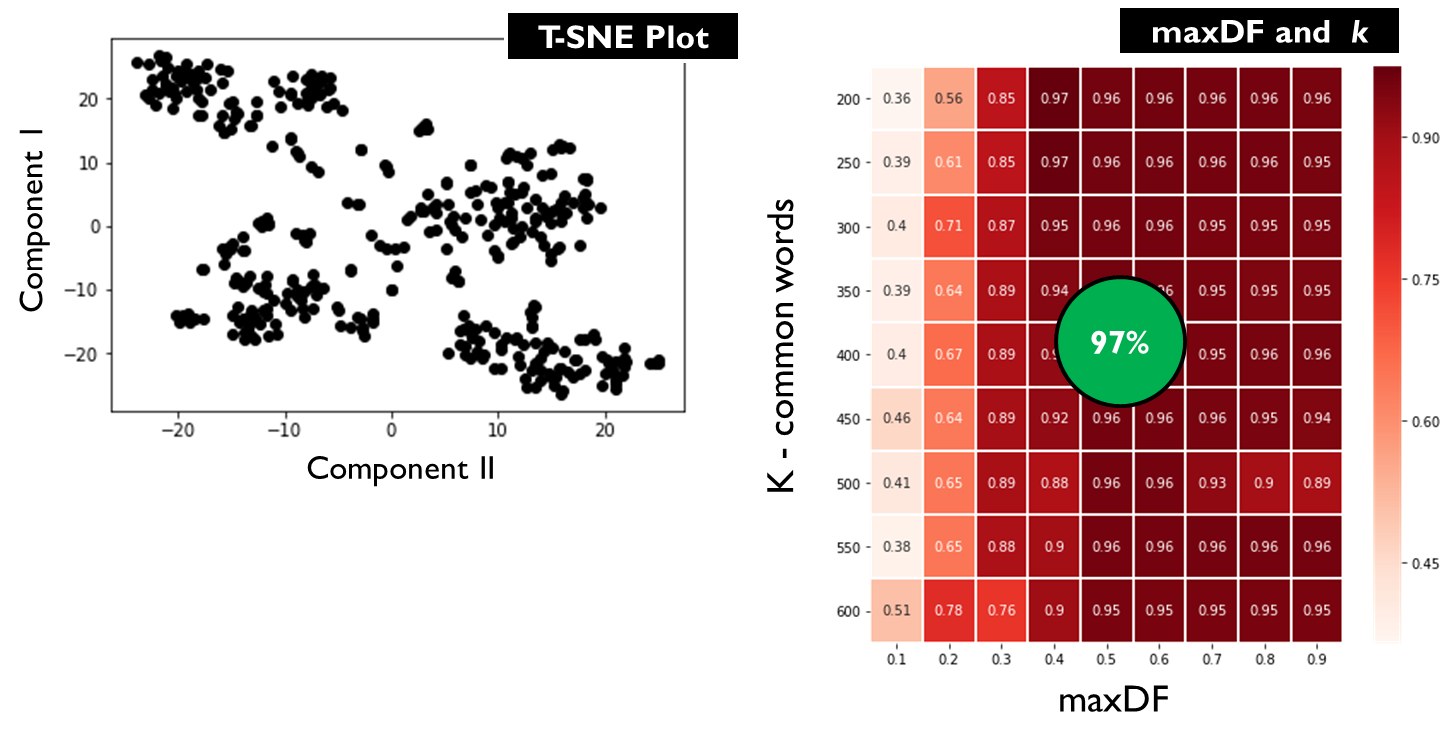


Figure 5. T-SNE plot (left) for Approach 2, showing 4 distinct clusters; Purity as a function of k and maxDF (right)

* 1. **Approach 3**. This approach also produced good results with a high purities of ~96% achieved at N>150 and k<250 . T-SNE plot also confirms the presence of 4 distinct clusters. Compared with Approach 2, we note that the purity scores were consistently >84% across all values of ***k,*** suggesting that using the industry sector name to extract the closest words using word2vec vectors allowed more relevant words to be selected (as opposed to just selecting for maximum **DF**).

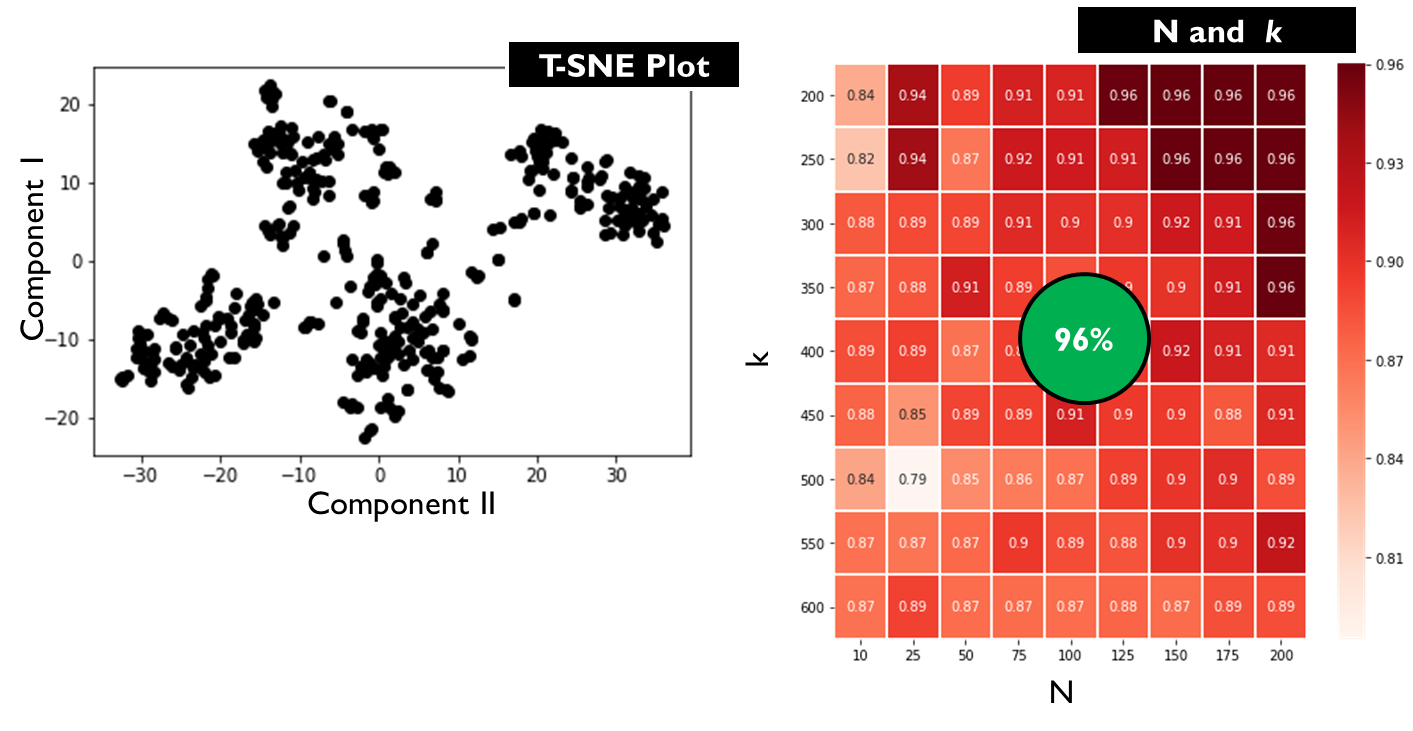


Figure 6. T-SNE (right) for Approach 3; Purity results as a function of N and k (right)

* 1. **Analysis.** We decided to focus on the TFIDF clustering model (Approach 2) as it was simpler and had fewer parameters to optimize. When visualized using a Tableau dashboard, it could be seen that Banking had a relatively lower purity score whereas the healthcare and real estate had a 100% purity score as seen from the T-SNE plot below (Fig.4). Our dashboard allowed further error analysis by displaying the top few words associated with each document when hovered over. **We found that Banking’s lower purity score was due to non-banking terms present in the document (this was not entirely surprising since banking services tended to cut across industry sectors). In contrast, the top few terms used in Real Estate and Healthcare documents tended to be domain specific (and hence document specific).** Fig.5 shows an example of a Real Estate document that was wrongly classified as a Banking document.

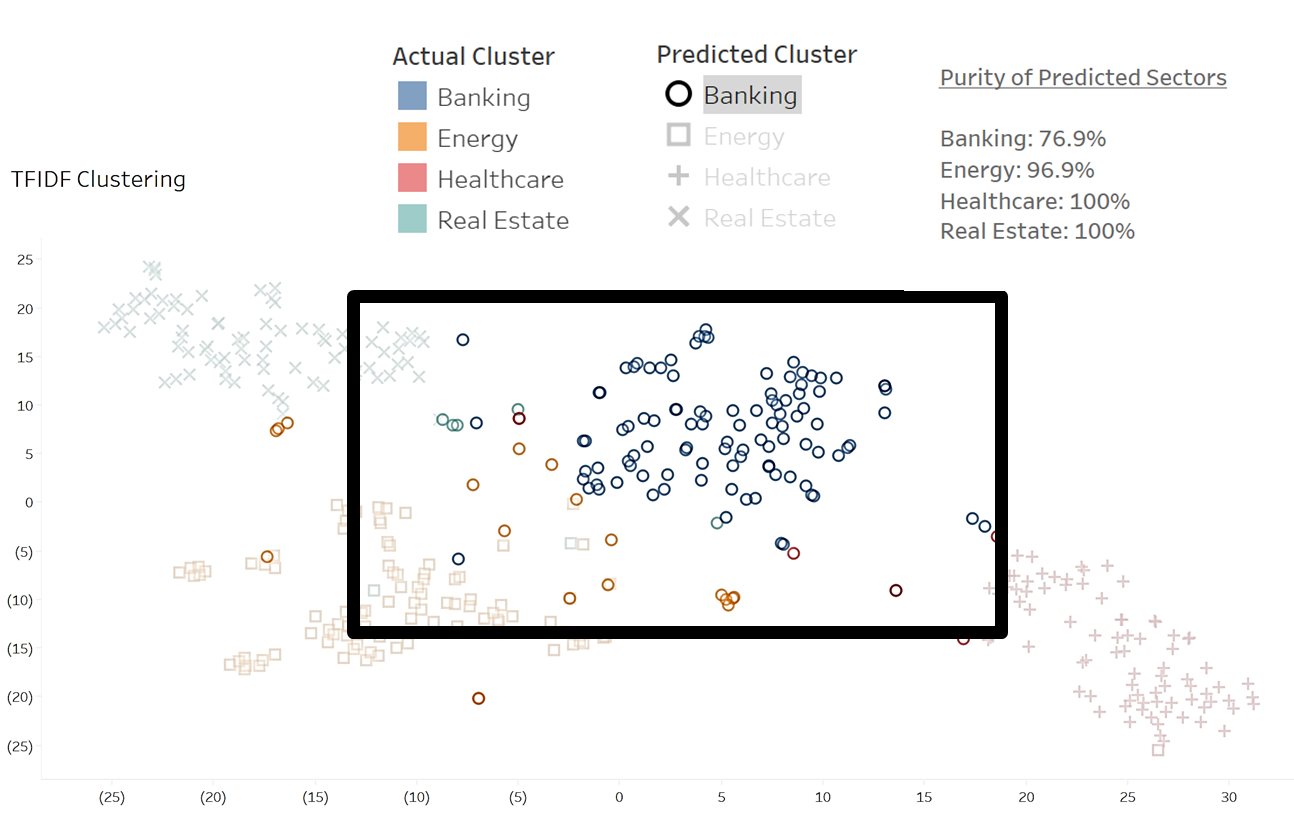


Figure 7. Tableau visualization of clustering results (with banking documents highlighted). True banking documents are in blue, while documents clustered as Banking are represented by circles. The presence of several non-blue circles indicate the Banking cluster was not as pure.

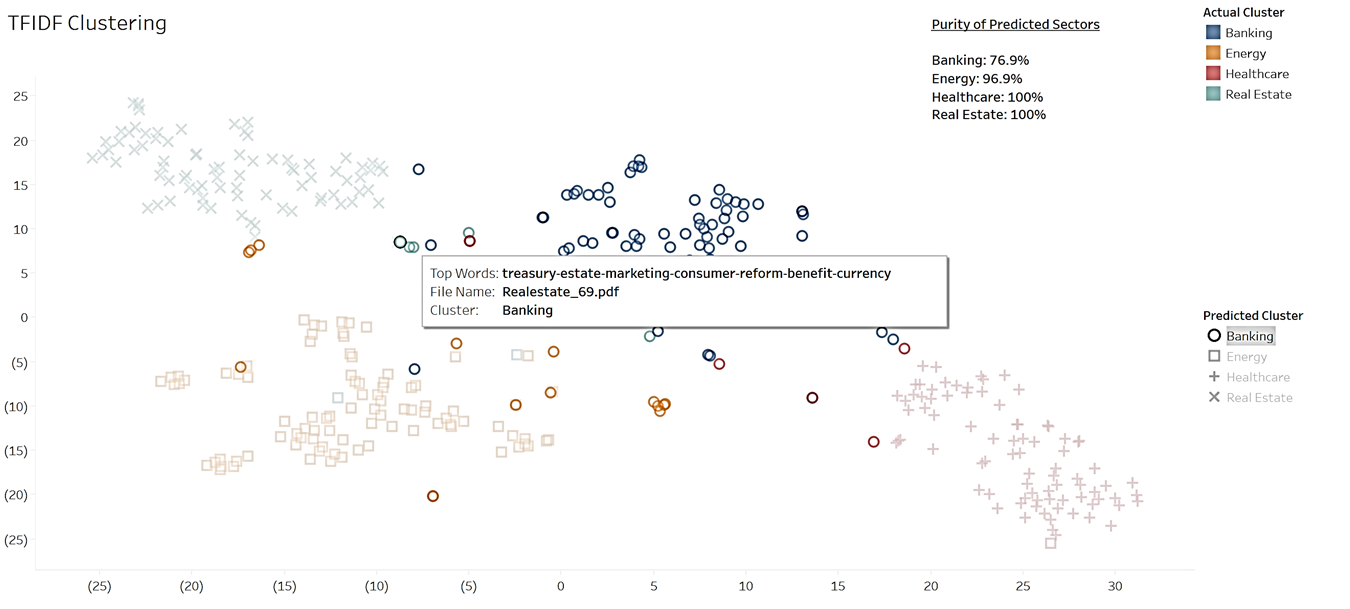
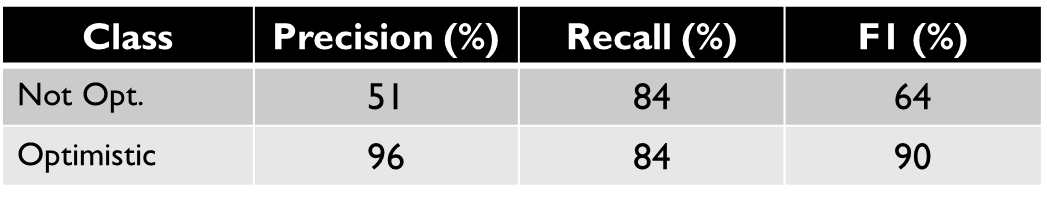


Figure 8. For one of the investment reports that was wrongly clustered into the banking clusters when it should be real estate instead, the top common words such as ‘treasury’ and ‘currency’ seems to be related to the banking sector as well.

1. **Sentiment Analysis**. Using logistic regression with our lexicon, we managed to achieve the results shown in Table 3. The overall accuracy (not shown) was 84% (note that we balanced the data set to contain equal numbers of both classes). Notably, we were able to achieve a high precision of 96% for the optimistic documents.

Table 3. Results of sentiment analysis using logistic regression.



1. Logistic regression also allowed us to examine which lexicon terms were more influential in the classification process. This could be observed by looking at the regression weights associated with each term (see Fig.9). It can be seen that the positive weights generally correspond to terms that convey optimistic sentiment (e.g. “Expansion”, “Attractive” and “Strong”), while negative weights corresponded to more cautious sentiment (e.g. “Downgrade”, “Sell”, “Modest”). **This result is intuitively pleasing, and suggested that the chosen lexicon reasonably reflected each document’s sentiment position.**

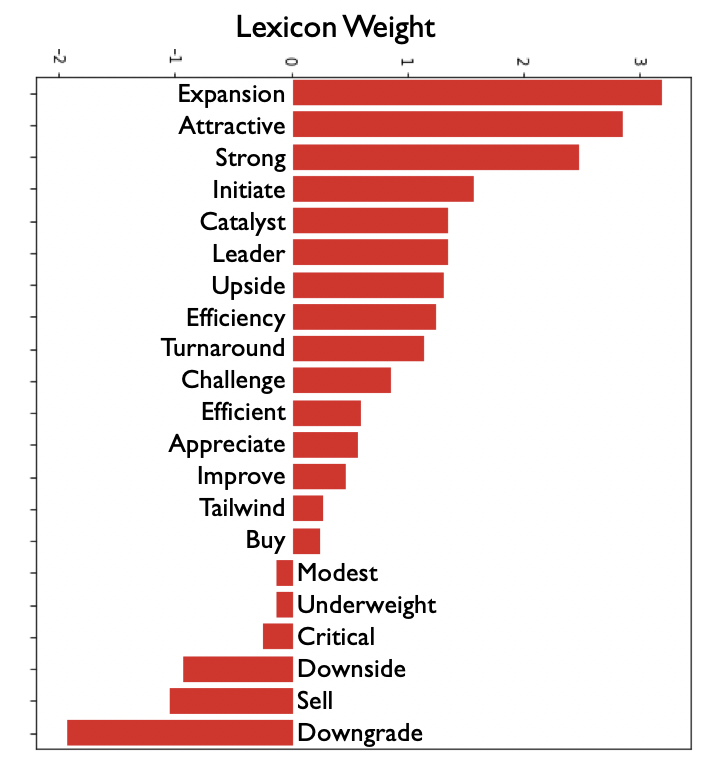


Figure 9. Logistic regression weights for each lexicon term. These terms only reflect the sentiment words that were present in this corpus. We note that there were fewer non-optimistic words than optimistic ones – possibly because there were much fewer non-optimistic reports to start with, or the vocabulary used to describe non-optimistic sentiment was more restricted.

1. **Visualization of Sentiment Results**. We also created a Tableau visualization to analyze the clusters of positive and negative sentiment reports. Within the 4 sectors, the positive bias (i.e. there were more optimistic reports than non-optimistic ones) is consistent throughout. We also showed the Term Frequency of the lexicon words on the bar chart at an aggregated level (Fig.10).

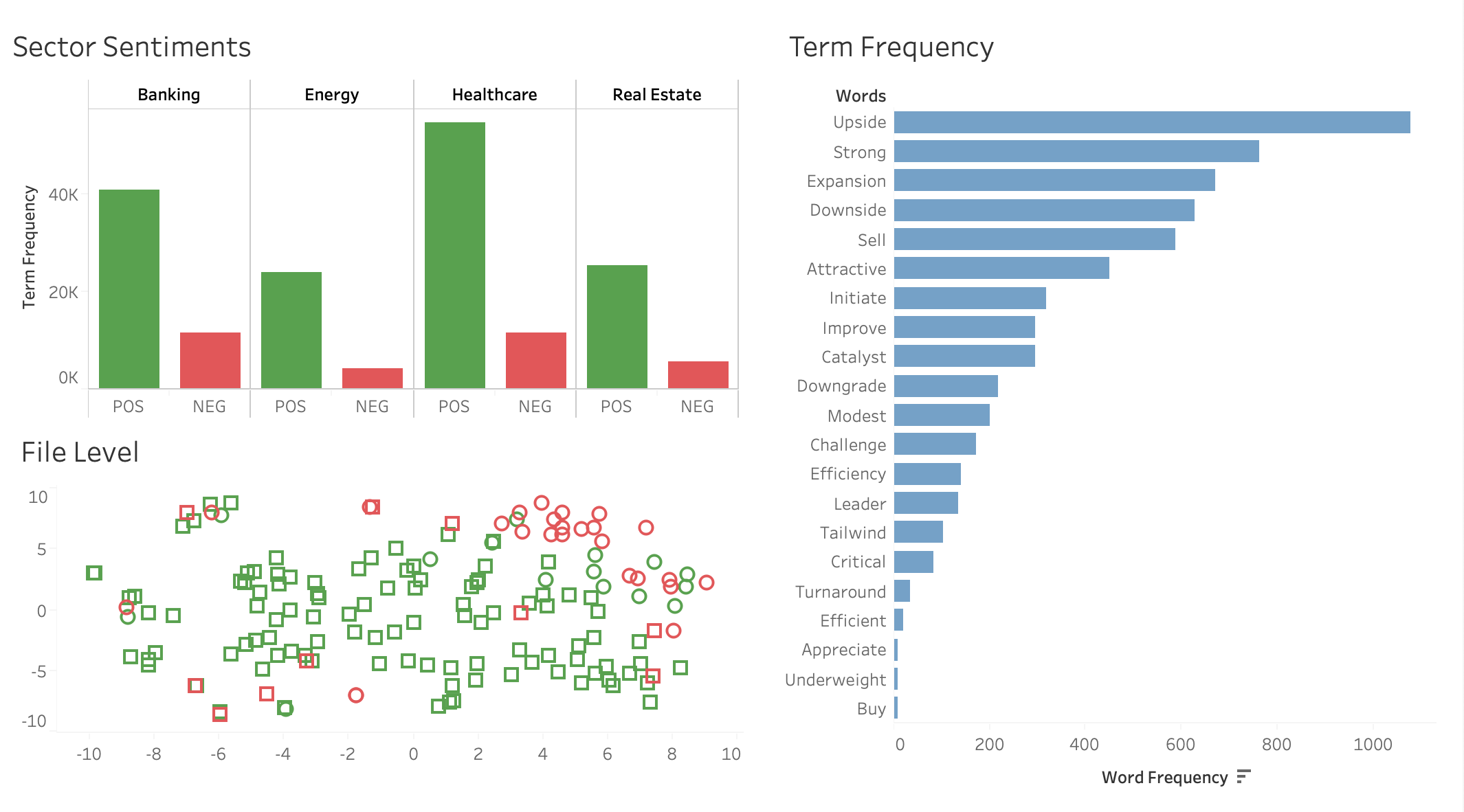


Figure 10. Tableau visualization showing the distribution of document sentiment as a function of the sector (top-left), and the aggregated frequency of the lexical terms (right). It also shows the T-SNE scatter-plot of the test-set files (bottom-left).

We can also see the accuracy of the predicted positive sentiment cluster, whereby the false-positives are the red squares. Note that there are many more green data points – this reflects the class imbalance in the test set.

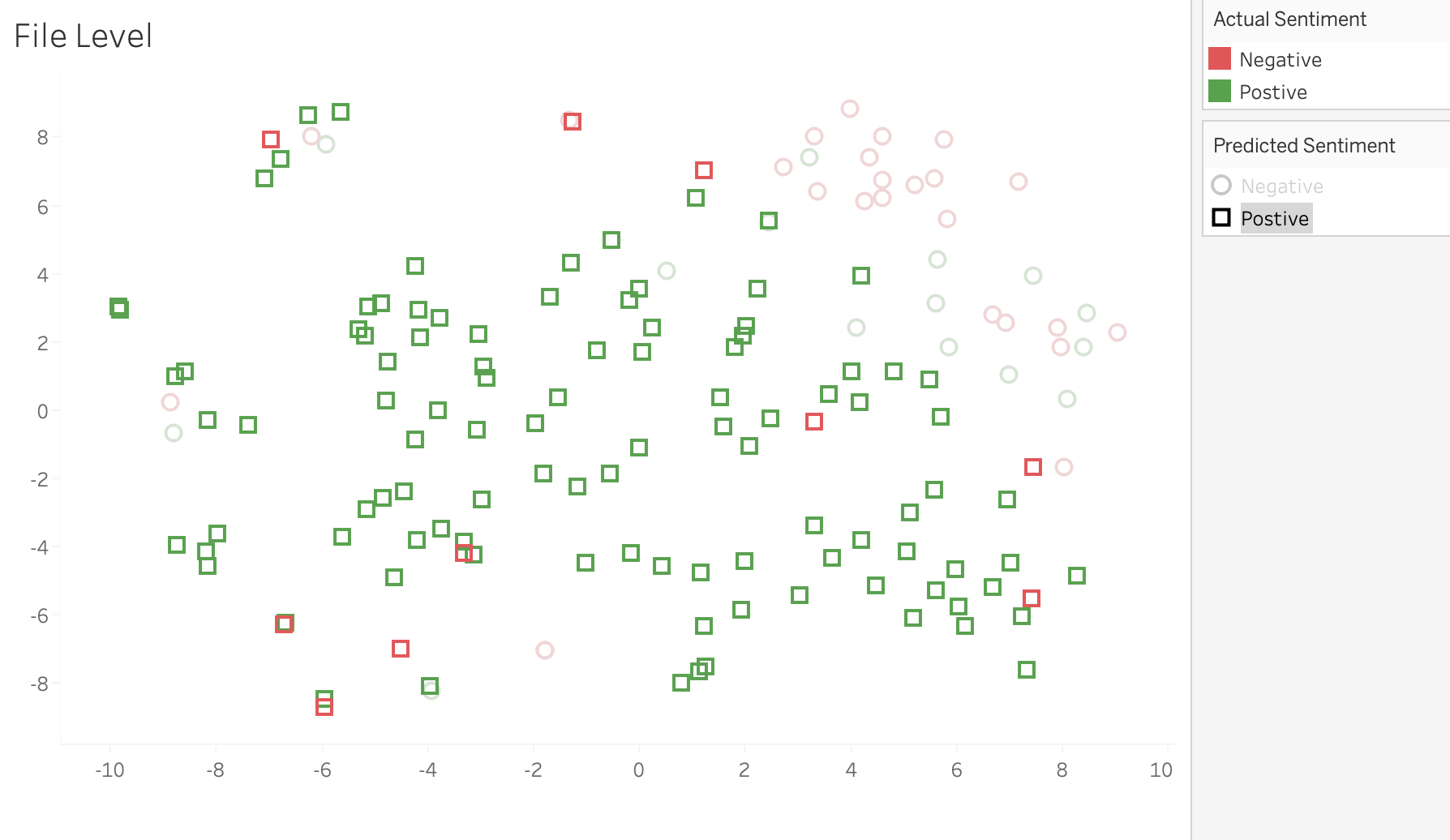


Figure 11. A closer look at the T-SNE plot for sentiment with optimistic sentiment documents highlighted (in green squares). Red squares indicate reports that had non-optimistic sentiment, but were wrongly classified as optimistic.

1. When we narrow in by file level on an individual file that is **correctly predicted as not-optimistic**, we can see that the negative words such as ‘sell’, ‘downgrade’, ‘downside’, overwhelm the positive words such as ‘upside’, ‘improve’, and ‘strong’ to give the report an overall negative classification (Fig.12). This is expected for a report that is non-optimistic.

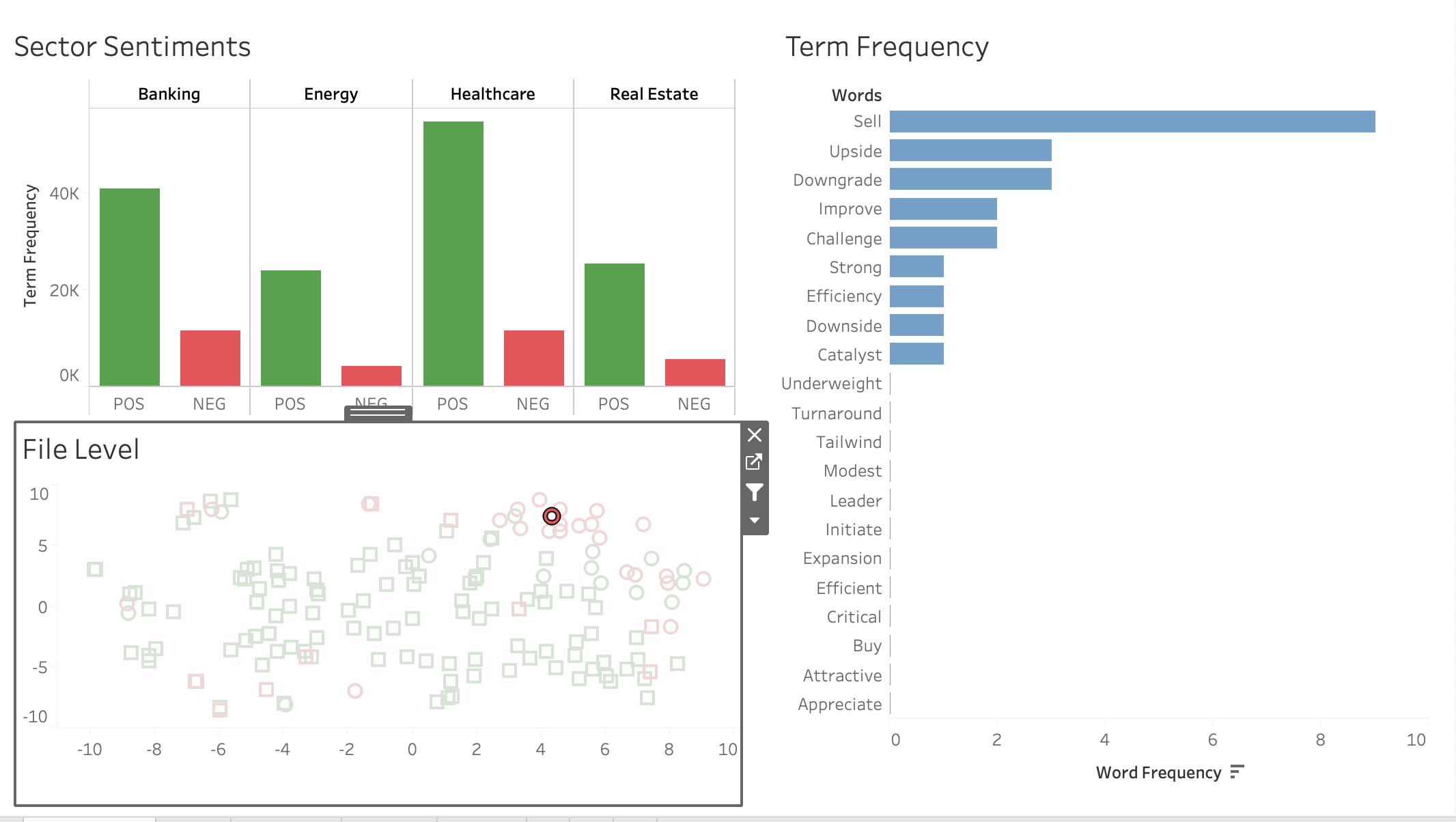


Figure 12. Example of a correctly classified report that was not optimistic. Dominant words such as ‘Sell’ and ‘Downgrade’ dominated the regression weights.

1. When we narrow in by file level on an individual file that is **correctly predicted as optimistic**, we can see that the positive words such as ‘strong’, ‘expansion, ‘upside’, overwhelm the negative word such as ‘negative’. There is also only 1 negative term here vs 6 positive terms which gives the report an overall positive classification.

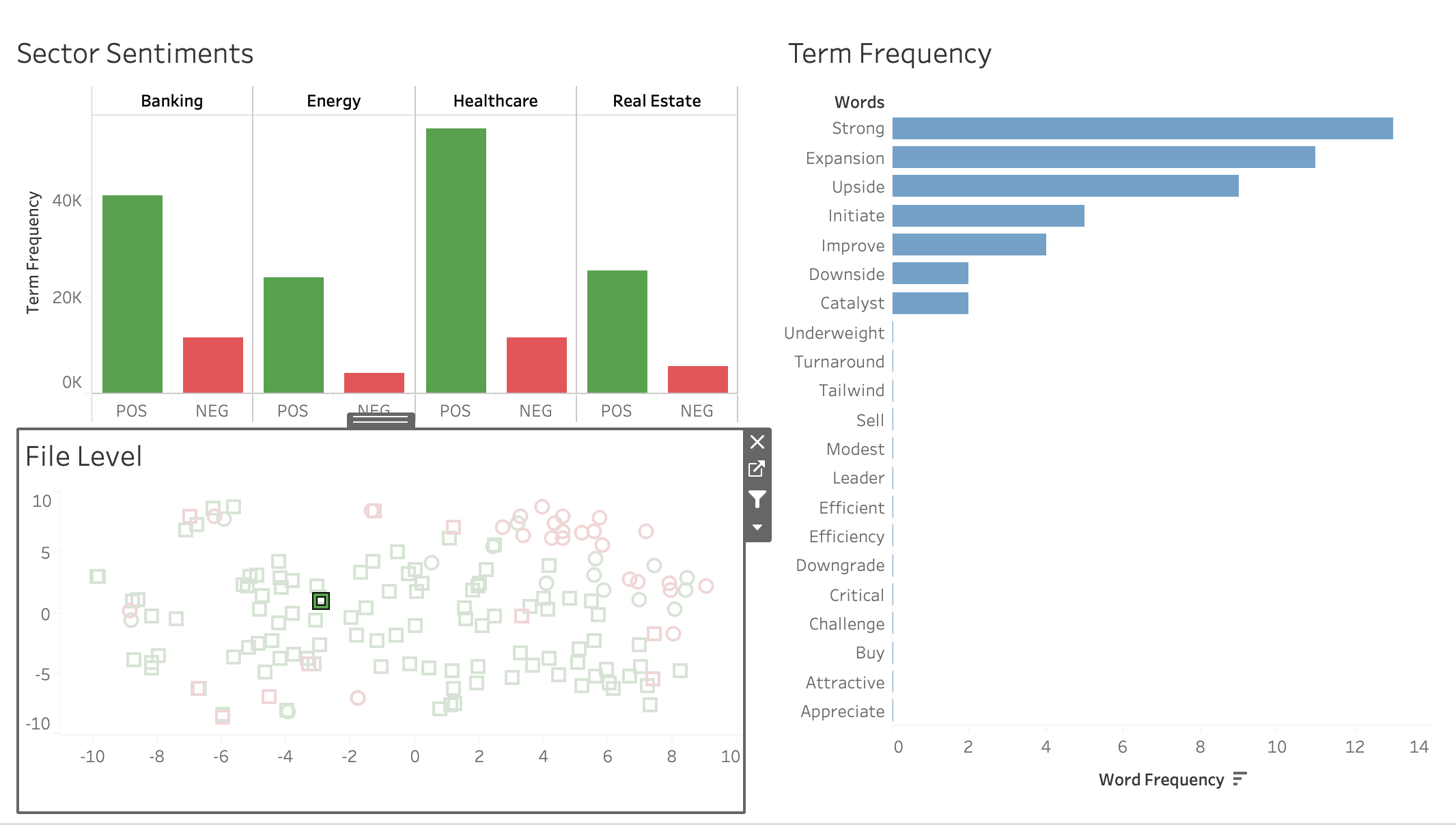


Figure 13. Example of a correctly classified report that was optimistic. Dominant words such as ‘Strong’ and ‘Expansion’ dominated the regression weights.

1. **Outcomes for Sentiment Analysis**. For a research analyst, knowing whether the overall sentiment is positive for the sector he is writing his recommendation report on **will help give him the context as to whether he is bucking the current industry trend and will help to justify and provide further empirical evidence to back his call.** For traders, this will help them understand what the current market is feeling and provide awareness to market changes for hedging their position exposure accordingly.
2. **NER for Optimistic Reports (only featured in the demonstration)**. Here, we implemented a NER function to extract company names from optimistic reports (using healthcare reports as an example). The NER function was taken from the Spacy library. In our implementation, the NER function would search for tokens with an “ORG” tag which returned names of organizations. An example is shown in Fig.14. The NER results were not 100% accurate as many of the lesser-known companies which were not part of spaCy’s organization list were not able to be captured. Nevertheless, **this function completes our demonstration of the solution, allowing traders to zero-in on specific companies of interest.**

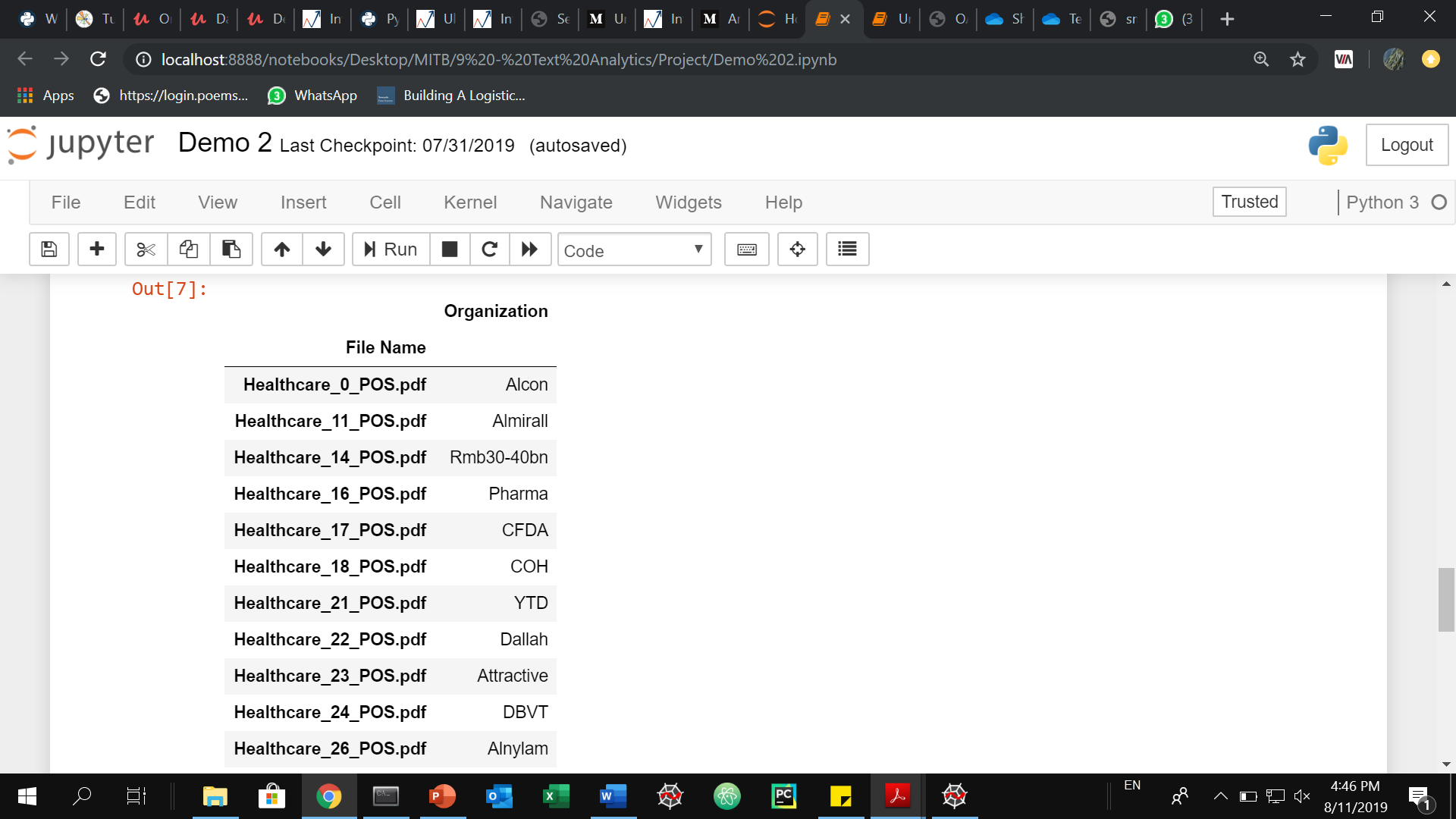


Figure 14. Extracting names of organizations from positive-sentiment reports. The File Name column corresponded to the original PDF file from which the organization name was extracted. If the user is keen to follow up, he/she can easily access the original PDF for more details.

**Discussions and Gap Analysis**

1. The following points are raised for discussion.
   1. **Securing the documents**. The team had intended to sign an NDA and obtain the actual daily investment reports from a local bank’s Equities department to perform the clustering and sentiment analysis. However, due to the internal bank policy restrictions, we were not able obtain the reports. This was despite initial efforts to showcase a proof of concept (using sample financial reports) to illustrate what the final solution could do. **As such, the team used financial reports from an online source.**
   2. **Deriving clusters with good purity results.** Initial attempts (without any feature selection/engineering) at clustering yielded rather poor purity results. Upon examining the top words in the files across the 4 sectors, the team discovered that many common words, which are not sector specific, appeared frequently in all the sectors. Examples of these were words like ‘investment’, ‘global’, ‘risk’, ‘potential’ and ‘dividend’. In this context, we regarded these terms as stopwords. After these words were then added to the stopwords list for removal. **Clustering performance improved drastically. Considerable domain knowledge was required to decide on the stopwords.**
   3. **Quality of source data.** Most of the financial reports used to train the model for sentiment analysis presented an optimistic outlook on the economy and/or sector. This was a prevalent phenomenon in the underlying sentiment of investment houses’/banks’ financial reports. To improve the overall accuracy and precision of the non-optimistic class, we would need more non-optimistic samples. In this implementation, **we recognize that the training data had less than 90 non-optimistic samples, which could have been insufficient to precisely classify the non-optimistic documents.**

## **Conclusion**

1. The team developed a solution to rapidly cluster and classify investment research reports, allowing traders to quickly categorize reports into their industry sectors and determine their investment sentiments. **The proposed solution will save the users time from manually classifying >1000 reports into respective sectors and sentiment groups. Based on the sentiment analysis that has been performed on the reports, business users are able to distinguish reports that are of interest. Coupled with the next step of information extraction, promising companies in different sectors can be quickly identified to help business users expedite their decision-making process.**

**Proposed Future Work**

1. Here, we propose several extensions that could augment the performance and functionality of the existing work:
   1. **Improving clustering purity**. The project can be further extended by adopting a more supervisory approach when dealing with documents from the banking sector since it has the lowest purity score among the 4 sectors that have been explored. This is because the words used to describe the banking sector are common to the rest of the sector. **A possible approach is to extend the keywords used for clustering by looking into bi-grams and/or trigrams** instead of unigrams alone. Possible improvements in key-word specificity using bigrams include examples like “bank loan” vs “bank”, or “interest rates” vs “rates”.
   2. **Improving Sentiment Analysis accuracy**. Another area that can be considered is to enhance the sentiment analysis on the documents. Currently, the project is only looking into two categories. However, it has been identified as a challenge since some documents contain both optimistic and non-optimistic terms, making it difficult to determine the appropriate category they should belong to. **A possible improvement is to introduce a third category (e.g. neutral) to capture these documents with mixed sentiments**. At the same time, additional functions can be implemented to allow users to flag out documents with mixed-sentiments, and prompt for user intervention, so that human judgement could be applied to verify if they should be parked in another category instead.

**References**

[1] Dhruvil Karani, (2018, Sep 1), Introduction to Word Embedding and Word2Vec, <https://www.tensorflow.org/tutorials/representation/word2vec>

[2] Susan Li (2018, Aug 17), Named Entity Recognition with NLTK and SpaCy, <https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da>

1. In the financial markets and investment community, people usually speak about investment or market sentiment. This sentiment is usually referred to as the amalgamation of participant’s attitudes in the investment world. They are usually referred to as ‘bullish’ (or optimistic), and ‘bearish’ (or not-optimistic) on the other hand [↑](#footnote-ref-2)
2. Briefly, the **Skipgram** configuration takes every word in the corpus (target word), and each target word’s surrounding context words (within a defined window) to then feed into a neural network. The network will predict the probability for each word to appear in the window around the target word. Once the probability has been maximized for each target-context pair, the weights associated with the neural network’s hidden layer can be seen as the dimensions of the word vector. In practical terms, word2vec provides us with a means of representing each word in a reduced vector space, where cosine similarity between the vectors are able to capture similarity in word meanings. [↑](#footnote-ref-3)
3. T-SNE (T-distributed Stochastic Neighbor Embedding) is a non-linear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space. [↑](#footnote-ref-4)