CS230

Classify Large Corporation's Industries based on their Descriptions to Identify Critical Investment Verticals for Various Industries

King Castillo Alandy Dy*
School of Engineering, Individualized Major
Stanford University
iamking@stanford.edu
Video Presentation: https://www.youtube.com/watch?v=Ize2RssFtAE&feature=youtu.be

Abstract

I have a list of companies (not classified by vertical) and their investments. I also have another list of companies with just their description and classification. I want to find out which verticals energy companies are investing in to say where corporations foresee the future of the industry. Are they investing in companies in the same verticals, complimentary verticals or completely different verticals.

1 Introduction

We should know where large companies in various industries are investing their money so organizations like the government can prepare to support these trends in the private sector. When we look at Bloomberg or other online resources to check what incumbents are investing in, the vertical these incumbents are in is unclear. It is unlabeled because incumbents operate in many verticals. We can use the descriptions of companies to predict and classify what verticals they specialize in. Through the research company Orbis, we purchased a dataset of human labeled companies and their descriptions. These companies and company names do not match the companies online. That's why we need to train a classifier for labeling what verticals the incumbents are in with NLP on their descriptions.

The input to our algorithm is a company name and its description on online news sites. First we use TFIDF. We test it on a bunch of classifiers and also two neural networks just to compare the accuracy between both.In this specific scenario, we are doing the classification to select the Energy vertical for corporate incumbents. This allows us to simplify the problem into a binary classifier which can just be retrained for other classifications when you purchase those datasets from Orbis. (College students don't have $$$) Afterwards, as a bonus, we will visualize the data we collect. In the future, this is like one of many binary classifiers and combine all into one larger model where we end it with a multiclass softmax layer.

2 Related work

There is a lot of work related to the space of classifying words based on just text. For example, a paper that really influenced my approach was Convolutional Neural Networks for Sentence Classification by Yoon Kim. In fact, the CNN I used in my second attempt is pretty much the same but with an extra layer added on given the complexity inherent in the problem I am working on. For this problem specifically as I detail later, there is a lot of patterns that need to be caught between words far away from each other and there are multiple ways in which every word may be used so seeing broader patterns is important. Other works that are relevant are included as references.

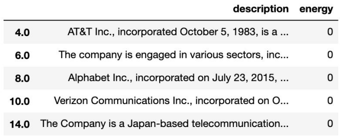
3 Dataset and Features

Table A - Database of energy companies and their descriptions

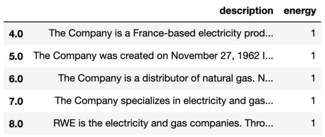
Table B - Database of non-energy companies and their descriptions

Table C - Database of unclassified companies with their descriptions and the companies they invest in

Used TF-IDF to take into consideration frequency and importance of various words normalized by how often they appear in relevant and irrelevant texts. We removed stop words, stemmed the words, and deleted all the irrelevant parts of the data. (In all company descriptions that contained the company's history, it said it after the string "HISTORY:" so we omitted everything else each time we saw this substring. We got our training data from OrbisResearch which is a leading market research reseller. The unlabeled data is from PitchBook, another data seller.



	description	energy
4.0	AT&T Inc., incorporated October 5, 1983, is a ...	0
6.0	The company is engaged in various sectors, inc...	0
8.0	Alphabet Inc., incorporated on July 23, 2015, ...	0
10.0	Verizon Communications Inc., incorporated on O...	0
14.0	The Company is a Japan-based telecommunication...	0



	description	energy
4.0	The Company is a France-based electricity prod...	1
5.0	The Company was created on November 27, 1962 I...	1
6.0	The Company is a distributor of natural gas. N...	1
7.0	The Company specializes in electricity and gas...	1
8.0	RWE is the electricity and gas companies. Thro...	1

To the left are samples of the data. We then do the pre-processing we previously described. People often use bigrams or trigrams but we weren't exactly sure how it would perform for this specific dataset. That's why we decided to select hyperparameters by graphing the accuracy depending on n-grams and the number of features being used. We took the one with the highest accuracy. More of this is described in the methods section.

The features themselves were selected using TF-IDF so we could select the most relevant features for our classifier. We identified the optimal number of features and n-grams. We have 18,677 labeled samples. For our training set, we used 95% (17,743) and for our validation set, we used 5% (934). Approximately ~80% are tagged as 0 and the rest are tagged as 1. We have no issue with the disparity in the number because this ratio of 0s and 1s also applies to the data we are running the classifier on. We have 49,685 unlabeled data points. An example of a description post-processing looks like this: "compani creat novemb 27, 1962 it oper seven divisions. the sale segment focus sale electr gas product servic end users..."

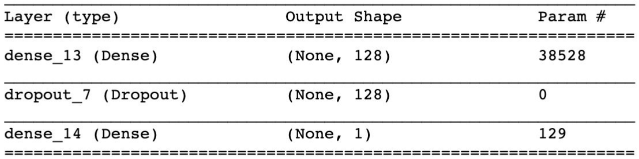
4 Methods

In our report we used multiple learning algorithms. I'll only explain the best performing non-deep learning solution here. For the passive aggressive classifier, we start by initializing the weights to zero and we predict positive if the sum of the weights multiplied by the features is greater than zero. There exists a buffer between -1 and 1 such that if the algorithm result or prediction is in that buffer region, we penalize the algorithm and modify the weights and same obviously goes for a wrong prediction. This puts a big buffer region between the predictions making it more certain of its choices. This can be interpreted mathematically as this loss function.

$$\sum_{j\neq y_i} \max(0, s_j - s_{y_i} + 1)$$

We then update the weight vector (theta) by adding y (whether we add or subtract is dependent on whether y is 1 or -1) and multiplying that by the loss by the features.

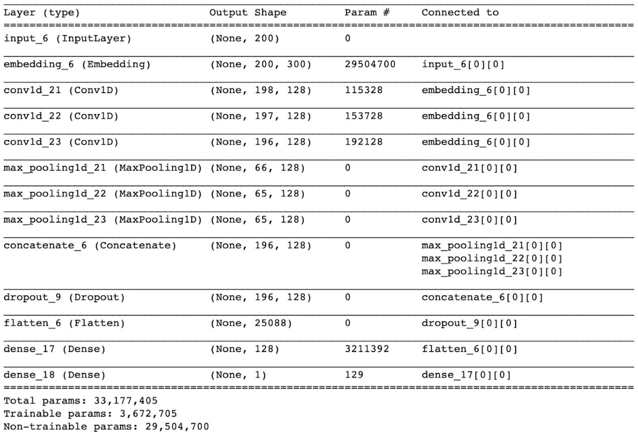
Here is the first model architecture I tried. It is very shallow.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_13 (Dense)             (None, 128)               38528     
_________________________________________________________________
dropout_7 (Dropout)          (None, 128)               0         
_________________________________________________________________
dense_14 (Dense)             (None, 1)                 129       
=================================================================
```

I experimented with different dropout rates here. Even at 0.5, there was a significant difference between training accuracy and testing accuracy so I was overfitting to the data. When I increased the dropout rate all the way to 0.8, then the difference became more reasonable. I used sigmoid since it was a binary classification problem.I performed mini-batch gradient descent with a batch_size of 40 and at 30 epochs. The optimal accuracy was hit by the 3rd and 12th epoch so it wasn't necessary but I let it run since it was going pretty quickly anyways. With all of this we were able to achieve: Training accuracy: 0.9788, Testing accuracy: 0.9711

It trained extremely fast and experimented a lot with the different batch sizes. It was interesting to see how drastically it would change the time it took to compute. I felt like I could have doubled or tripled the batch size but when I would run it, it still didn't make a significant difference in the accuracy.



```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to                     
==================================================================================================
input_6 (InputLayer)            (None, 200)          0                                            
__________________________________________________________________________________________________
embedding_6 (Embedding)         (None, 200, 300)     29504700    input_6[0][0]                    
__________________________________________________________________________________________________
conv1d_21 (Conv1D)              (None, 198, 128)     115328      embedding_6[0][0]                
__________________________________________________________________________________________________
conv1d_22 (Conv1D)              (None, 197, 128)     153728      embedding_6[0][0]                
__________________________________________________________________________________________________
conv1d_23 (Conv1D)              (None, 196, 128)     192128      embedding_6[0][0]                
__________________________________________________________________________________________________
max_pooling1d_21 (MaxPooling1D) (None, 66, 128)      0           conv1d_21[0][0]                  
__________________________________________________________________________________________________
max_pooling1d_22 (MaxPooling1D) (None, 65, 128)      0           conv1d_22[0][0]                  
__________________________________________________________________________________________________
max_pooling1d_23 (MaxPooling1D) (None, 65, 128)      0           conv1d_23[0][0]                  
__________________________________________________________________________________________________
concatenate_6 (Concatenate)     (None, 196, 128)     0           max_pooling1d_21[0][0]           
                                                                 max_pooling1d_22[0][0]           
                                                                 max_pooling1d_23[0][0]           
__________________________________________________________________________________________________
dropout_9 (Dropout)             (None, 196, 128)     0           concatenate_6[0][0]              
__________________________________________________________________________________________________
flatten_6 (Flatten)             (None, 25088)        0           dropout_9[0][0]                  
__________________________________________________________________________________________________
dense_17 (Dense)                (None, 128)          3211392     flatten_6[0][0]                  
__________________________________________________________________________________________________
dense_18 (Dense)                (None, 1)            129         dense_17[0][0]                   
==================================================================================================
Total params: 33,177,405
Trainable params: 3,672,705
Non-trainable params: 29,504,700
```

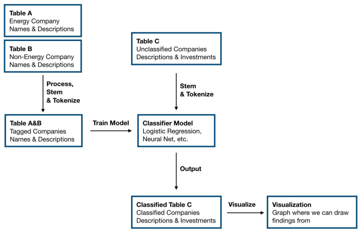
People often assume that deeper neural networks will generally perform better. In this case, it is definitely true. It was much slower but also quite significantly more accurate. I mimicked https://arxiv.org/pdf/1408.5882.pdf Yoon Kim's CNN but I used sigmoid for the last sense layer because it is a binary classifier and I had more than 1 dense layer. I added this dense layer because I think there are longer more complex patterns in being able to identify what a specific type of company is. For example, a company might say:

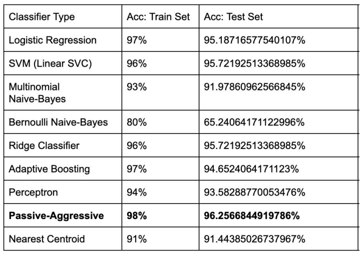
‘Green’, ‘field’, ‘pastur’, ‘provid’, ‘people’, ‘livestock’
If we only notice simpler patterns like whether or not the words green and provide are there, this could easily be mistaken as an energy company. However, by knowing that when provide and livestock are in the same sentence, it should be able to pickup that it probably is not a provider of electricity through a deeper neural network which can understand more complex patterns. These changes were made through trial and error and thankfully brought us to an even higher accuracy than I thought was possible. At the moment, I am overfitting but I increased the dropout to 0.7 and it actually decreased my accuracy significantly for both the test and the train accuracy. I reverted back to 0.6 with the deeper architecture which I set up and this allowed me to achieve an accuracy of 0.9732.

5 Experiments/Results/Discussion

First step is to preprocess all data. We want to remove stopwords, stem and tokenize the data to make sure that it is ready for feeding into our model.

We took randomly took 95% of Table A&B and used this as our training set. The rest was used for the validation set. We first ran TF-IDF. Unigrams seemed to be the best option for this since the neural net would pick up on more complex patterns afterwards anyways.





Classifier Type	Acc: Train Set	Acc: Test Set
Logistic Regression	97%	95.18716577540107%
SVM (Linear SVC)	96%	95.72192513368985%
Multinomial Naive-Bayes	93%	91.97860962566845%
Bernoulli Naive-Bayes	80%	65.24064171122996%
Ridge Classifier	96%	95.72192513368985%
Adaptive Boosting	97%	94.6524064171123%
Perceptron	94%	93.58288770053476%
Passive-Aggressive	**98%**	**96.2566844919786%**
Nearest Centroid	91%	91.44385026737967%

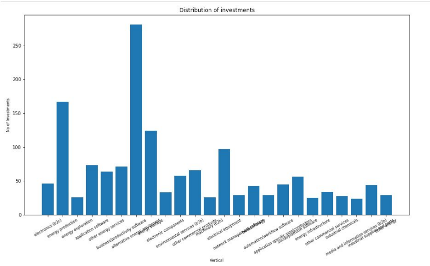
After selecting the hyperparameters that would represent the data best, it was time to try it on different models.

For almost all of them, the accuracies and F-Scores on the training set were just 1 or 2% higher so we do not have a high bias or high variance problem. We ran the algorithm on the unclassified data and it came up with 497 companies out of 49685 as 1. The rest were classified as 0. Manually looking through around a hundred datapoints in classification 1, all of them were correctly classified. We would still prefer that the set of selected companies would be greater in number but we would much rather have it have zero false positives. The rationale for this is because this sample of 497 companies is large enough to generalize for the rest of the investors in this area.

Above are all of the non deep learning classifiers we tried. With deep learning, it is far more accurate and impressive.

6 Conclusion/Future Work

To summarize, we have a dataset of labeled and unlabeled companies and their respective descriptions. We trained different classifiers and deep learning models to do this. In the future, if we have a more computationally powerful machine, we could try an even deeper neural network though I’m not sure how significantly this would improve the product. Some of the classifications to be made are pretty subjective and I feel like I would only get around 98% right myself based on



a small sample size where me and my friend cannot agree on whether or not a company can be considered in the green space.

Long term, we can create several classifiers which will then feed their output into a multiclass softmax layer that then identifies which class it most likely fits into.

This segment is not really part of this ML class but this last part is about visualizing the output of the classifier to show our results. The verticals that Energy companies invest in the most would be productivity software, energy production and lastly alternative energy equipment.

References

[1] Sebastiani, Fabrizio. "Machine learning in automated text categorization." ACM computing surveys (CSUR) 34.1 (2002): 1-47.

[2] Miškuf, Martin, and Iveta Zolotová. "Comparison between multi-class classifiers and deep learning with focus on industry 4.0." 2016 Cybernetics & Informatics (K&I). IEEE, 2016.

[3] Louw, Abraham, et al. “Global Trends in Renewable Energy Investment 2018.” Global Trends in Renewable Energy Investment 2018.

[4] http://www.iberglobal.com/files/2018/renewable_trends.pdf

[5] This research is by Iberglobal and it analyzes the investments being made in the space of energy. It does not research on the exact same question but is very similar in that it is looking if most of the investments being made are in research, government projects, startups, etc. Instead of the specific vertical, it is researching on the way it makes the investment. The techniques used here were completely manual hand-labeling techniques. They used no Machine Learning. The benefits would be more accurate data. The detriment is that they did not label as much data as us since we used an algorithm to do it.

[6] Sebastiani, Fabrizio. "Machine learning in automated text categorization." *ACM computing surveys (CSUR)* 34.1 (2002): 1-47.

[7] This research is from way back in 2002. We extend upon a lot of the techniques used in this research and apply them to our specific use case. There were a lot of IR techniques used like TF-IDF and data preprocessing techniques like getting stem words and removing stop words. Like our paper, they compared and contrasted SVMs, ensembles, and many other types of classifiers. Really, our research is a direct application of their research into the business and investment sector. We are using essentially the same learning algorithms which they discussed in this research paper.

[8] Miškuf, Martin, and Iveta Zolotová. "Comparison between multi-class classifiers and deep learning with focus on industry 4.0." *2016 Cybernetics & Informatics (K&I)*. IEEE, 2016.

[9] In this research they compared multi-class classifiers and deep learning classifiers for industry applications. Again, our research paper is a specific implementation of various techniques used here. We, however, were more focused on specific NLP techniques and implement a more diverse range of classifiers for the purpose of this research.