Machine Learning Engineer Nanodegree

Capstone Project

Weblink: https://www.notion.so/Machine-Learning-Engineer-Nanodegree-02ac730c8e284d3394beab2bdf69dd9e

Iain MacCormick

June 9th, 2020

I. Definition

Project Overview

With a wealth of information and opinion available in real time through news, social media and investor publications there is a vast source of financial textual data, which if understood correctly can be used by investors and traders to better understand companies and markets.

This project will attempt to classify the the sentiment of a Financial statements from the point of view of a retail investors. Since sentiment of financial news articles and headlines, as well as other financial texts, can be an indicator of company stock returns and volatility it is important for retail investors to have access to an overview of sentiment from the financial news towards a company.

Sentiment analysis is an algorithmic approach used to understand the emotional context of a body of text and is used in a wide range of domains including but not limited to: understanding polarity of financial texts, interpreting product reviews and identifying racism in social media posts.

The dataset that will be used for experimental analysis is known as the "Financial Phrase Bank" dataset containing sentences of positive, negative and neutral sentiments. In some of the approaches covered, domain specific dictionaries will be used.

Problem Statement

The aim of this project is to give retail investors access to statistics describing the sentiment of financial news headlines for companies of their interest. To achieve this functionality a Sentiment Classifier will need to be trained, tested and deployed and a web application will need to be built to enable the user to retrieve predictions.

Modelling Steps:

1. Download dataset and conduct exploratory analysis.
2. Conduct a series of experiments with various feature engineering steps and classification algorithms, discussed in algorithms and techniques section.
3. Select the final model based upon performance against our selection criteria discussed in the metrics section.

Web Application:

The selected model will be deployed and integrated within web application to give retail investors access to sentiment analysis for news relating to companies of interest. The purpose of this document is to cover the experimental analysis only however the final application can be accessed here: https://fin-sent.herokuapp.com/

Metrics

We aim to minimise the number of true positives and true negatives so accuracy will be used to evaluate the benchmark and solution model. Precision, Recall and

F-Score will be used alongside accuracy to help understand the performance of the model during development. As the problem is a multi-class classification with class imbalance the micro averaging method will be used for these metrics.

Gradient boosting involves three elements:

1. A loss function to be optimized.
2. A weak learner to make predictions.
3. An additive model to add weak learners to minimize the loss function.

All metrics will be cross-validated using the K-Fold cross validation technique.

$$Accuracy = \frac{\sum_{i=0}^{n}(TP_i + TN_i)}{\sum_{i=0}^{n}(TP_i + TN_i + FP_i + FN_i)} = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \forall i$$

$$Precision = \frac{\sum_{i=0}^{n} TP_i}{\sum_{i=0}^{n}(TP_i + FP_i)} \qquad Recall = \frac{\sum_{i=0}^{n} TP_i}{\sum_{i=0}^{n}(TP_i + FN_i)}$$

$$Fscore = \frac{2 * Precision * Recall}{Precision + Recall}$$

$$n : number\ of\ classes$$
$$TP_i : True\ Positives\ class\ i$$
$$TN_i : True\ Negatives\ class\ i$$
$$FN_i : False\ Negatives\ class\ i$$
$$FP_i : FalsePositives\ class\ i$$

II. Analysis

Data Exploration

The dataset that will be used for experimental analysis is known as the "Financial Phrase Bank" dataset. It contains around 5000 sentences described by "Sentiment" which classifies the sentence as being positive, negative or neutral. Each sentiment was chosen by highest level of agreement across 16 annotators, with each annotator coming from a business education background. **[1]** Examples:

- Neutral: "customers in a wide range of industries use our stainless steel and services"
- Negative: "pretax profit decreased to eur33.8m from eur40.8m in the fourth quarter of 2005"
- Positive: "diluted earnings per share rose to eur0.52 versus eur0.09"

Domain specific dictionaries are used, during the tagging stage of Approach 1. All dictionaries described below were provided on request by the author of **[2]**

- Leading performance indicators: words which represent drivers of company's future performance. Examples include "operations", "partnership", "stores" and "deal".

- Lagging performance indicators: words which represent the consequence of company's activity. Examples include "gdp", "roi", "profit_before_tax" and "eps".
- Directionality: words which represent direction of movement. Examples include "rocketed", "win-win", "reductions" and "decreased."
- Finance specific sentiment words: words indicating positive of negative financial sentiment. Examples include "outperform", "surpass", "burden" and "decline".

Exploratory Visualisation

Fig 1. describes the distribution of the sentiments present in the dataset with 60:30:10 being the approximate ratio between Neutral:Positive:Negative sentiments. Given this class imbalance, any learnt approach will need to ensure that it does not fit to the dominant classes.

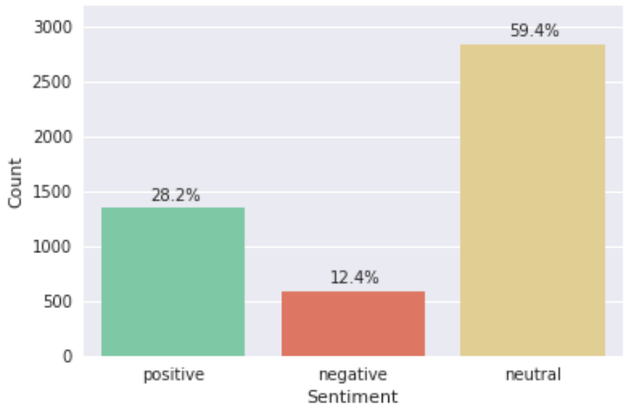

Figure 1. Distribution of sentiments in the financial phrase bank dataset.

The underlying dataset contains only one feature, the sentence to be analysed. To investigate the dataset further, analysis was conducted on features generated. Fig 2. describes the difference between the underlying sentiment distribution described in Fig 1. and the distribution of tags generated in the Lexicon Tagging feature creation step.

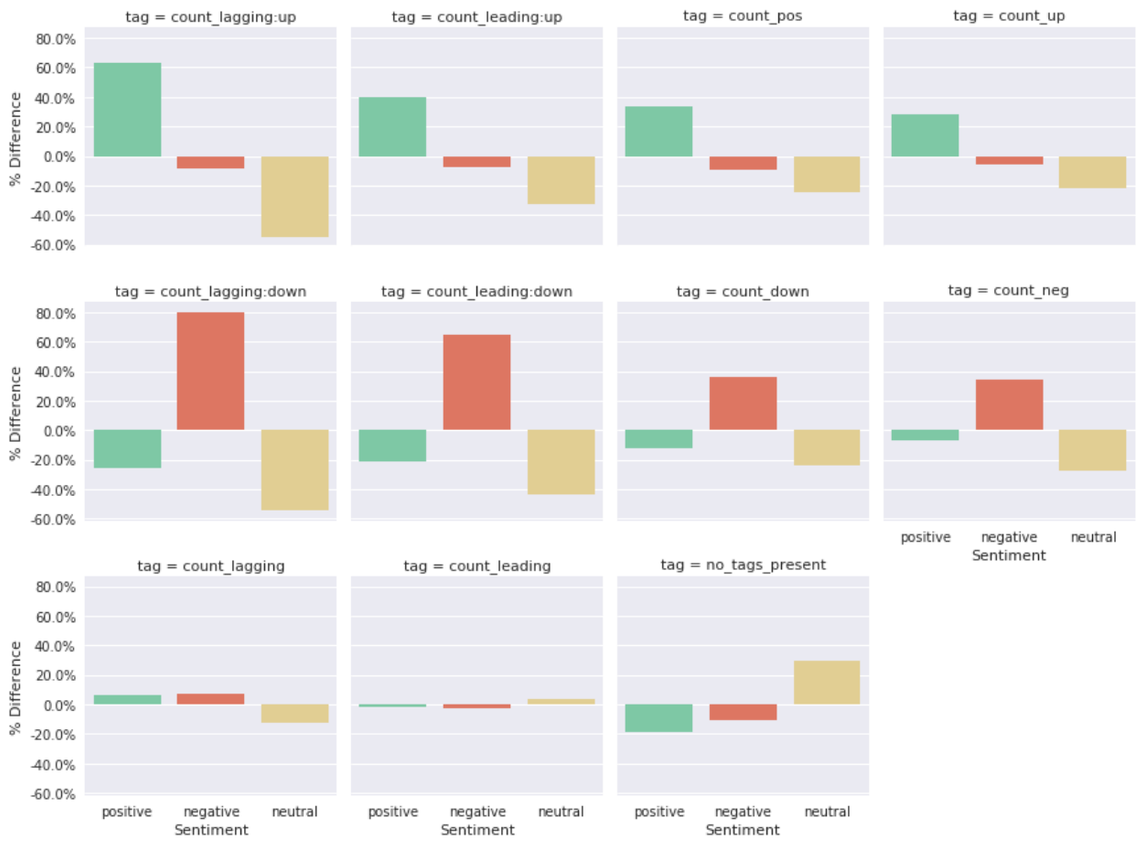

Figure 2. Percentage Difference between overall sentiment distribution and sentiment distribution of sentences containing given tags.

Fig 2. gives confirmation to the logic behind the tagging process which is centred around how a human may approach the problem. That is to say:

- Tags positively correlated with positive sentiment: lagging:up, leading:up, pos, up.
- Tags positively correlated with negative sentiment: lagging:down, leading:down, neg, down.
- Tags positively correlated with neutral sentiment: no tags present.
- Ambiguous tags: lagging, leading

Algorithms and Techniques

Across the experimental analysis phase a total of 7 experiments were conducted. Below we discuss the algorithms and techniques used in these experiments. They were picked due to their varying degree of explainability, progression of results and fundamental difference in methodology.

Approach 1 - Lexicon Tagging & Association Rule Mining

Lexicon Tagging

Each sentence is parsed and tagged if it contain words which are present in the lexicon. In addition, tags are created when a performance indicators are in context with directionality.

- "The lithuanian beer market was `up` 14.41 million litres in January, a `rise` of 0.8 percent from the year-earlier figure" — `up`, `up`

- In the second quarter production at the plant increased by 20% — leading:up

Table 1. Lexicon categories and corresponding tags used to tag sentences in the dataset.

Lexicon Category	Tags
Leading Indicator	leading
Lagging Indicator	lagging
Directionality	down up
Finance Sentiment Words	neg pos
Performance Indicator + Directionality	lagging:down lagging:up leading:down leading:up

Association Rule Mining

Association rule mining is a method which can be used to understand how items in a collection are related with each other. It is a method commonly used in retail to understand which products are frequently purchased together for instance at a supermarket. Definitions:

- Itemset: a collection of one or more items
- Support: the proportion of transactions in which the itemset X appear
- Confidence: the probability of finding itemset Y in transactions given the transaction already contains itemset X.
- Antecedent: the itemset which existed before or logically precedes the consequent itemset.
- Consequent: the itemset implied by the antecedent.

$$Support(X) = \frac{n_X}{n} \qquad Confidence(X \rightarrow Y) = \frac{Support(X \cup Y)}{Support(X)}$$

$$n : number\ of\ examples$$
$$n_X : number\ of\ examples\ containing\ X$$

The Apriori algorithm provides a method by which all rules (X→Y) exceeding a specified support threshold and confidence threshold are exceeded. This approach can be extended to classification by including the target variable within the itemset. All association rules generated which do not include the target variable as the consequent itemset are discarded.

Why might this approach work for financial classification?

This approach is based upon an algorithm proposed in the literature **[2]**. The lexicon tagged feature generated is closely aligned with how a human might interpret a financial text and therefore the predictions generated are also easily interpreted and analysed. It is a useful to gain deeper understanding of the problem before moving on to more sophisticated and less interpretable solutions.

Approach 2 - TFIDF & SVC

TFIDF

TFIDF refers to the combination of two probabilistic functions, Term frequency and Inverse Document Frequency. Term frequency rewards frequency within a given document and Inverse Document Frequency rewards rarity across a collection of documents. The outcome of combing these two metrics is to assign a high weight to discriminative words in a document.

SVC

A support vector machine finds a hyperplane, known as the decision boundary, that separates classes by maximising the orthogonal distance between the boundary and the points closest to the hyperplane, known as the support vectors. In the event there is no clear separation, a support vector machine undergoes a process known as kernelling which transforms the problem into a higher dimensional space until the classes become separable. There are different types of kernel which can be applied which define what we mean by separable e.g. linearly separable, polynomially separable. A parameter known as the soft margin can also be applied which allows for a tolerance of misclassification.

Why might this approach work for financial classification?

TFIDF could remove the need for a domain specific lexicon, which in turn removes the risk that such a lexicon does not adequately describe the underlying words which are able to recognise or make distinctions on sentiment with accuracy. SVCs have the potential to work well on text classification problems for three key reasons (summarised in detail in **[4]**)

1. SVCs can handle large feature space and document vectors, often have a large feature space.
2. SVCs can handle sparse inputs and document vectors, such as TFIDF, are often sparse.
3. Most text categorisation problems are linearly separable.

Approach 3 - CountVectorisedTags & GradientBoostingMachines

CountVectorisedTags

Equivalent to the Bag of Words approach but applied to the tags generated in the lexicon tagging feature creation step.

GradientBoostingMachines

Gradient boosting includes three fundamental elements: a loss function, a 'weak learner' to make predictions and an additive model to sum weak learners and minimise the loss function. Gradient descent is used to minimise the loss function, but rather than optimising parameters with respect to a loss function on each iteration a new weak learner is added to the model.

Why might this approach work for financial classification?

CountVectorisedTags in combination with gradient boosting machines were chosen due to their underlying difference to the other methods used so far, with the view to creating an ensemble of different methods.

Approach 4 - Ensemble

An ensemble method is used to take a collection of machine learning models and combining to obtain greater predictive power. The success of such approach is reliant on different underlying models learning different distributions within the data.

Benchmark

Since no official benchmark exists, a simple benchmark is proposed. The dataset is imbalanced so an approximate 60% accuracy can be obtained by assuming all headlines are neutral due to the sentiment distribution in the dataset.

III. Methodology

Data Preprocessing

The "Financial Phrase Bank" dataset contains sentence:sentiment pairs. There are no other features in the dataset so all preprocessing steps focus on the text in the sentence provided and can be summarised as followings:

1. All sentences converted to lowercase.
2. Removal of stop words.
3. Separation of numbers and units. For example eur3.2m becomes eur 3.2 m.
4. There are examples of where floating point numbers are separated with whitespace. For example eur0. 12m converted to eur 0.12 m.
5. Underscores added between phrase words in the lexicon. Phrases are multiword entries in the lexicon. For example "operating profit" is converted to "operating_profit" both in the lexicon and text.

There were two key feature creation steps which have been described:

1. Lexicon tagging
2. TFIDF

Implementation

Approach 1 - Lexicon Tagging & Association Rule Mining

The apriori algorithm is used to mine the itemsets and generate association rules. A pre-defined implementation of the apriori algorithm from the apryori python package was used.

The algorithm uses the association rules generated to make predictions on unseen examples by searching for exact matches and partial matches between the lexicon tag and the association rules and accumulating corresponding confidence values to each of the classes. More specifically, if the itemset (list of tags for unseen example), matches exactly with the entire antecedent itemset in the association rules, then predict the consequent class (sentiment). If no exact matches are found, iterate through each tag in the unseen example and match with an

antecdent itemset in the association rules, returning its corresponding cofidence. The predicted class will be the one with the highest average confidence accross all tags. If no partial matches are found the classes are assigned probabilities in line with the underlying sentiment distribution of the dataset. Table 2 and Table 3. provide examples of how predictions are made in the exact match, partial match and no match cases.

A vectorised implementation of this described method was written as iterative methods were not performant.

Table 2. Example of association ruleset generated through apriori algorithm.

type	antecedents	consequent sentiment	confidence
group of tags	lagging:down, neg	negative	1
group of tags	down, lagging	negative	0.95
group of tags	leading, pos, lagging	positive	0.9
single tag	leading:down	negative	0.85
single tag	leading:up	positive	0.8
single tag	positive	positive	0.75
single tag	negative	negative	0.7
single tag	no_tag_present	neutral	0.65

Table 3. Example of how predictions are made on unseen examples using association rules.

Tags	match type	Positive Conf	Negative Conf	Neutral Conf	Prediction
down, lagging	Exact	0	0.95	0	negative
positive, negative, leading:down	Partial	0.75	(0.7+0.85)/2=0.775	0	negative
positive, negative, leading:up	Partial	(0.75+0.8)/2=0.775	0.7	0	positive
lagging	None	0.3	0.1	0.6	neutral

Approach 2 - TFIDF & SVC

A pipeline was created using Sklearn's TF-IDF vectoriser and Sklearn's support vector classifier.

Approach 3 - CountVectorisedTags & GradientBoostingMachines

The unique tags generated in the lexicon tagging stage were count vectorised. In other words each unique tag e.g. "lagging:up", "down", "positive" etc. becomes its own feature where the entry for a given sentence is the number of occurrences of

that tag. For example, if a sentence was tagged "up, up, positive" this feature would have a count of 2 in the feature "up" a count of 1 in the feature "positive" and a count of 0 for all other features.

Sklearn's implementation of Gradient Boosting Machines was used.

Approach 4 - Ensemble

The final prediction obtained is the one with the highest underlying probability across the three approaches discussed. For example if model 1 predicts 'neutral' with a probability of 60%, model 2 predicts 'negative' with a probability of 65% and model 3 predicts 'positive' with a probability of 75% then the final predicted class will be 'positive.'

Refinement

Lexicon Tagg

Fig 2. Suggested that the presence of the presence of 'leading' and 'lagging' indicators as individual tags may decrease predictive power. In practice this proved not to be the case.

Hierarchical Classification

All the approaches were struggling to classify between Neutral and Polarised which can be seen in Table 4. with 22-25% of the error being attributed to errors classifying between neutral and polarised vs 0-3% of the error being attributed to errors classifying between negative and positive. This is intuitively obvious as the differences between a Neutral and Positive/Negative sentence could be much harder to distinguish than between a Positive and Negative sentence.

Table 4. Total error decomposed into neutral vs polarised error and negative vs positive error. Total Error = 1 - Accuracy.

	experiment	total error	error: neutral vs polarised	error: positive vs negative
1	ar	0.26	0.258	0.002
3	cvt & gbc	0.225	0.212	0.013
5	tfid & svc	0.241	0.222	0.019

A hierarchical classification approach was used to try and address this fact. The first classifier is trained to distinguish between neutral and polarised classes and the second, which takes all examples classified as polarised by the first classifier as input, distinguishes between positive and negative classes Fig 3.

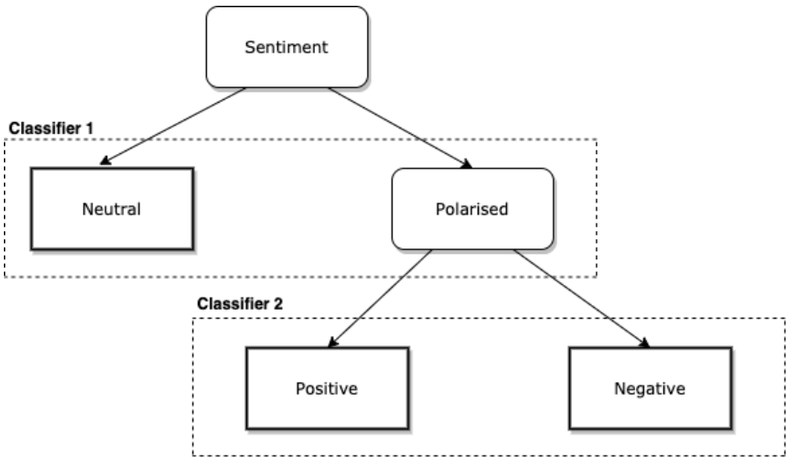

Figure 3. Hierarchical classification structure implemented.

This approach only improved reduced the error (neutral/polarised) in the association rules approach. For gradient boosting machines there was a negligible difference across all metrics between the flat classification and hierarchical classification approach and for the support vector machine approach hierarchical classification hindered all metrics.

Table 5. Total error of hierarchical classification approach, decomposed into neutral vs polarised error and negative vs positive error.

	experiment	total error	error: neutral vs polarised	error: positive vs negative
2	ar & hc	0.255	0.224	0.031
4	cvt & gbc & hc	0.234	0.22	0.015
6	tfid & svc & hc	0.252	0.229	0.023

Ensemble Method

One major problem with the hierarchical classification approach is that when misclassification occurs in the upper level of the hierarchy there is no way of the model correcting these errors in lower levels. It is therefore important that the upper level has as high accuracy as possible. As discussed above, the hierarchical classifier alone had limited success in increasing the accuracy of predictions in the neutral vs polarised case.

To address this, an ensemble method was proposed. Given there are only three candidates for each level of the hierarchy all combinations of algorithms were tested to obtain the most performant option.

IV. Results

Model Evaluation and Validation

Throughout the results section the following abbreviations are used:

- tfidf - term frequency inverse document frequency (Description, Implementation Details)
- cvt - count vectorised tags (Description, Implementation Details)
- ar - association rules (Description, Implementation Details)

- gbc - gradient boosted classifier (Description, Implementation Details)
- svc - support vector classifier (Description, Implementation Details)
- ens - ensemble (Description, Implementation Details)
- hc - hierarchical classification (Description, Implementation Details)

Model Validation:

To ensure model validity, 20% of the data was held back for the purpose of avoiding data leakage during hyper-parameter tuning. The results displayed in Table 6. were obtained using 5-fold cross validation on the remaining 80% of the dataset.

Progression of Results:

Table 6 shows the progression of results across all the experiments conducted. The algorithms generating the results in Table 6. were trained with default parameters specified in the appendix.

Table 6. Progression of results accross all experiments.
Results shown by accuracy in descending order.

	experiment	accuarcy	fscore	precision	recall
0	ar	0.734	0.65	0.779	0.605
1	ar & hc	0.745	0.708	0.695	0.728
2	tfid & svc & hc	0.761	0.7	0.735	0.678
3	cvt & gbc	0.775	0.728	0.762	0.708
4	cvt & gbc & hc	0.776	0.727	0.764	0.707
5	tfid & svc	0.779	0.724	0.77	0.696
6	hc ens	0.8	0.752	0.797	0.725

Final Solution Structure and Hyperparameters:

Fig 3. describes the structure of the hierarchical classifier which in the case of the final solution contains two ensemble classifiers. The first distinguishes between neutral and polarised classes and the second takes all polarised predictions and distinguishes between positive and negative classes. Each ensemble method is made up of two or more algorithms, which have been discussed, and the exact algorithms and their hyperparameters can be found in the appendix.

There was a significant improvement when an ensemble was created of a subset of the models. Table 7. shows a 2.2%-5.2% increase in accuracy and a 3.1%-5.05% increase in f-score from the underlying models to the ensemble model. Table 8. shows that the gain in overall accuracy is due decreased error in predicting neutral vs polarised sentences, the ensemble was created specifically to try and improve this tier of the hierarchical classifier.

Table 7. Results of ensemble classifier and underlying algorithms.

	experiment	accuarcy	fscore	precision	recall
0	ar & hc	0.745	0.708	0.695	0.728
1	tfid & svc & hc	0.761	0.7	0.735	0.678
2	cvt & gbc & hc	0.776	0.727	0.764	0.707
3	hc ens	0.8	0.752	0.797	0.725

Table 8. Total error for ensemble method and underlying algorithms. Decomposed into neutral vs polarised error and negative vs positive error.

	experiment	total error	error: neutral vs polarised	error: positive vs negative
0	ar & hc	0.255	0.224	0.031
1	cvt & gbc & hc	0.234	0.22	0.015
2	tfid & svc & hc	0.252	0.229	0.023
3	hc ens	0.211	0.192	0.019

Choosing Hyperparameters for the Final Model:

The underlying models suffered from a relatively low recall vs high precision, shown in Table 6, in particular this was caused by a large number of True Positives classified as Neutral. To improve the ensemble one or more of the underlying models needed to perform better in this area. The underlying models in the ensemble were tuned to try and balance precision and recall, without significantly impacting accuracy, in attempt to address this problem. If at least one of the models could perform better in this area, this could improve the performance of the ensemble as a whole. Fig 5. Fig 6. Fig 7. show the effect of model hyperparameters on the underlying algorithms' performance metrics.

Figure 5. Effect of hyperparameter values on precision, recall and accuracy metrics for AR.

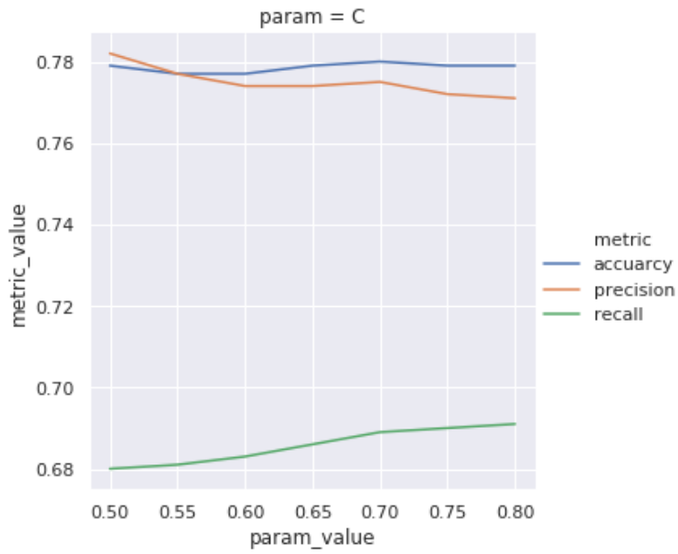

Figure 6. Effect of hyperparameter values on precision, recall and accuracy metrics for SVC.

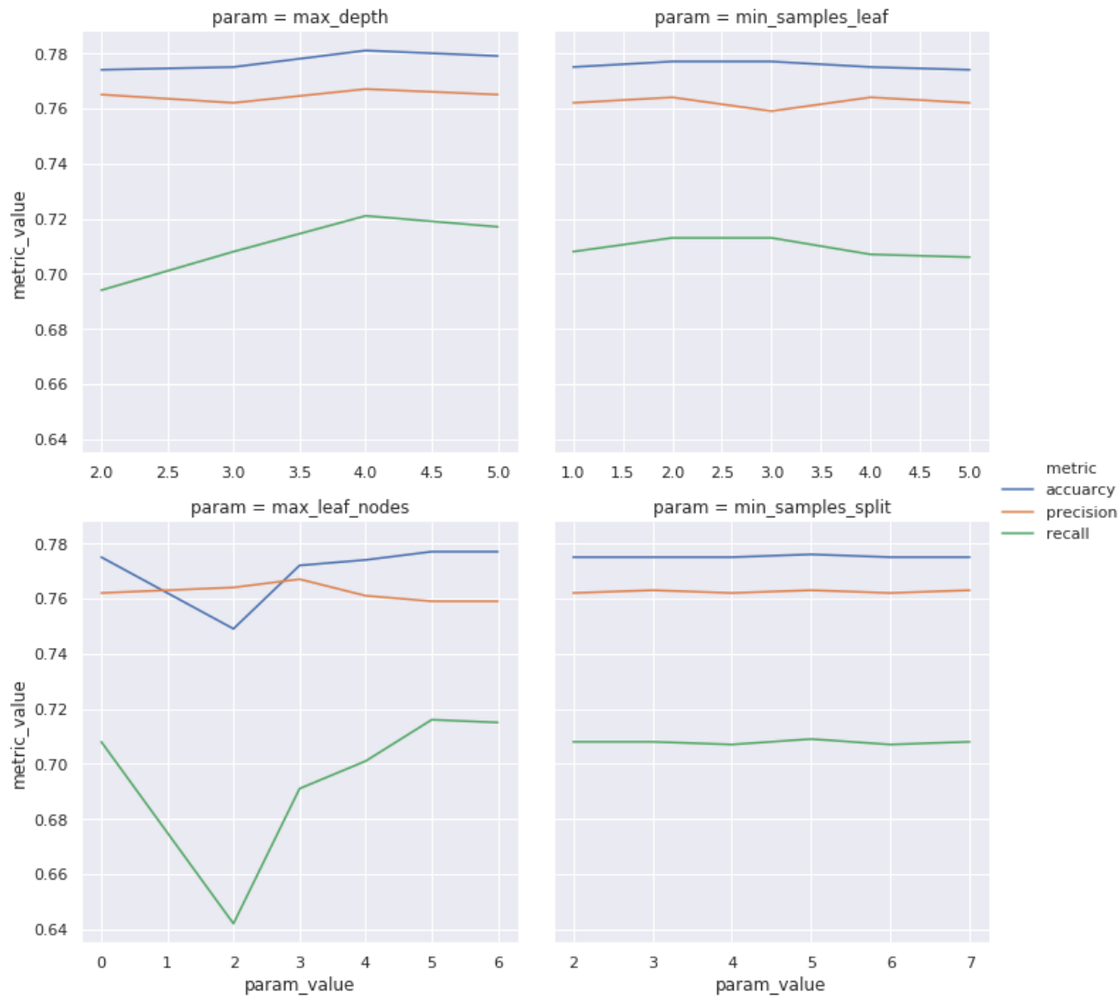

Figure 7. Effect of hyperparameter values on precision, recall and accuracy metrics for GBC.

Justification

The final model showed a 20% increase in accuracy compared to the benchmark proposed. In addition the range of results showed a 6% increase in accuracy and a 10% increase in Fscore from the worst to best performing model.

V. Conclusion

Free-Form Visualisation

Fig 7. shows the number compares the number of misclassified sentences by algorithm looking at where the different algorithms overlap. It shows that 81 sentences were misclassified by all three algorithms, 144 were misclassified by exactly two of the algorithms and 179 were misclassified by only one algorithm. This diagram led to idea to create the ensemble method, as it showed that the different techniques were learning different ways to approach the problem.

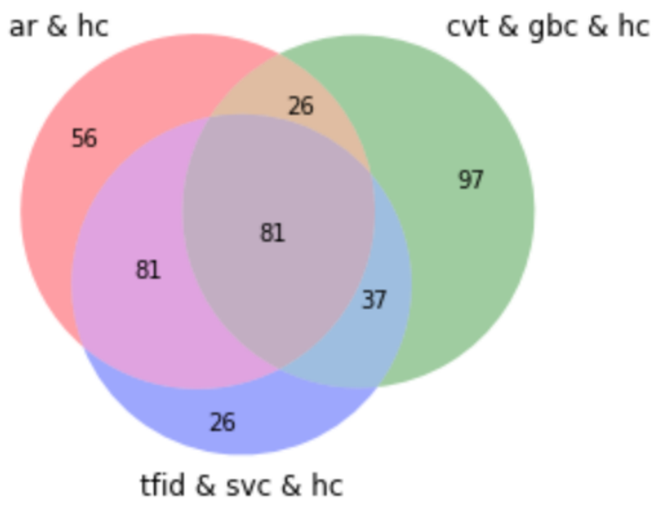

Figure 7. Venn Diagram showing overlapping miss-classifications between different algorithms.

Reflection

The final solution involves hierarchical classification, an ensemble method and three underlying algorithms. In particular the ensemble of algorithms was used to maximise the accuracy of the first tier of a hierarchical classifier which distinguished between neutral and polarised reviews, the most challenging part of the problem. The algorithms were effective within the ensemble as each individual algorithm and corresponding preprocessing steps 'understood' the problem from a different perspective.

A challenging aspect of the implementation was to build custom association rules classifier, hierarchical classifier and ensemble classifier which had to work in conjunction with one another, alongside sklearn methods.

Generally speaking, text classification can be particularly challenging, as understanding the errors and grouping them into meaningful sets to understand where a method is failing is a challenging task.

Although not discussed in this document, deploying the final solution, generating predictions on unseen examples and integrating this into a web application was a challenging and interesting part of the overall project. The final web application can be accessed here: https://fin-sent.herokuapp.com/. A username and password can provided on request.

Improvement

The association rules algorithm was the weakest member of the ensemble, however there are three potential improvements which could be made to this

approach during the lexicon tagging step:

1. The lexicon tagging step could be extended to look for number patterns within the text e.g. from $8m to $10m would be tagged 'up.' There were some examples in the misclassifications where such an approach would have resulted in a 'performance indicator:directionality' tag which often had strong predictive power.
2. Furthermore, the leading and lagging indicators could be split to create four groups where positive and negative leading lagging indicators are separated. For instance losses:increased would be tagged lagging:up which would infer a positive result.
3. Leading indicators are often industry specific, so this part of the lexicon could be expanded.

In future I would consider a deep learning approach for such a problem however it was not considered due to a limitation to the number of examples available in this dataset. In particular I would be interested to see how BERT, a transfer learning method for NLP, works against this dataset particularly in classifying neutral vs polarised examples.

VI. Other

References

[1] Pekka Malo⋆, Good Debt or Bad Debt: Detecting Semantic Orientations in Economic Texts — https://arxiv.org/pdf/1307.5336.pdf

[2] Srikumar Krishnamoorthy, Sentiment Analysis of Financial News Articles using Performance Indicators, 2017 — https://arxiv.org/pdf/1811.11008.pdf

[3] Tim Loughrabn and Bill McDonald, When Is a Liability Not a Liability? — https://www.uts.edu.au/sites/default/files/ADG_Cons2015_Loughran McDonald JE 2011.pdf

[4] Thorsten Joachims, Text Categorization with Support Vector Machines: Learning with Many Relevant Features — https://www.cs.cornell.edu/people/tj/publications/joachims_98a.pdf

Appendix

Default Model Parameters:

- AR: min_support=0.005, min_confidence=0.6, min_lift=0.0
- SVC: (penalty='l2', loss='squared_hinge', dual=True, tol=0.0001, C=1.0, multi_class='ovr', fit_intercept=True, intercept_scaling=1, class_weight=None, verbose=0, random_state=None, max_iter=1000)
- GBC: loss='deviance', learning_rate=0.1, n_estimators=100, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=3, min_impurity_decrease=0.0, min_impurity_split=None, init=None, random_state=None, max_features=None,

verbose=0, max_leaf_nodes=None, warm_start=False, presort='auto', validation_fraction=0.1, n_iter_no_change=None, tol=0.0001

Final Selected Model Parameters:

Ensemble Classifier, distinguishing between Neutral and Polarised sentiments, consisting of the following:

- Algorithm 1: TFIDF & SVC, Hyperparameters: C=1.5 (other params as default)
- Algorithm 2: CVT & GBC, Hyperparameters: max_depth = 4,min_samples_leaf = 3,max_leaf_nodes = 5, min_samples_split = 5 (other params as default)
- Algorithm 3: AR, Hyperparameters: min_confidence=0.5 (other params as default)

Ensemble Classifier, distinguishing between Positive and Negative sentiments, consisting of the following:

- Algorithm 1: TFIDF & SVC, Hyperparameters: C=1.5 (other params as default)
- Algorithm 2: CVT & GBC, Hyperparameters: max_depth = 4,min_samples_leaf = 3,max_leaf_nodes = 5, min_samples_split = 5 (other params as default)