Putting a Fine-Tuned BERT Model to the Test: Practical Implications of FinBERT

James Gao
University of California, Berkeley
School of Information
jamesygao55@berkeley.edu

Ajit Barhate
University of California, Berkeley
School of Information
ajitbarhate@berkeley.edu

Abstract

In this report, we present our findings when implementing a sequence classification task upon a Fine-Tuned BERT model using language from the field in which the BERT model was trained and tuned upon. Whereas traditional BERT covers a general language domain since it was trained using Wikipedia and the Book Corpus, FinBERT fine tunes upon BERT by doing additional training on financial documents and therefore covers more of a business and finance language domain. The builders of FinBERT have proven that it was able to achieve significantly more accurate results in MLM and NSP tasks than traditional BERT on financial documents. Our work builds on top of this and implements FinBERT as a sequence classifier to see if it is also able to achieve similar improvements compared to BERT. We take 10-Q reports, financial documents that publicly traded companies are required to file to the SEC each quarter, as our data set. Our classification labels are a reflection of how the stock price of each company performs immediately following the filing of the 10-Q report, acting as a type of sentiment reflection of how positive and confident the report made investors feel. With this implementation, we are exploring both the improvements that FinBERT is able to provide, as well as explore if there are truly significant signals in the 10-Q filings as it relates to stock price.

1 Introduction

Over the past few years, significant work and progress has been made to fine tune traditional BERT to specific language domains. As the familiarity with BERT and fine-tuning it continues to grow within the NLP community, more and more sophisticated models have been developed and published. The most well known ones are arguably BioBERT (Jinhyuk Lee and Kang, 2019) and SciBERT (Iz Beltagy and Cohan, 2019). As one of the first domain specific fine tuned BERT Models, BioBERT used additional biomedical focused corpora to continue training the model weights with using three different tasks: Named Entity Recognition, Relation Extraction, and Question Answering. Similarly, SciBERT was further trained on science related documents using some of the sames tasks that BioBERT used as well as Text Classification and Dependency Parsing. Other popular fine-tuned models include ClinicalBERT (Kexin Huang, 2020) and PatentBERT (Lee and Hsiang, 2019), which develop a better understanding of clinical notes and patent law, respectively.

The one that we are particularly interested in here is FinBERT (Vinicio DeSola1 and Nonis, 2019), implemented by recent University of California, Berkeley, School of Information Master's students. In a similar fashion to the other fine tuned BERT models, FinBERT was trained on finance specific documents, specifically, 10-K annual financial reports that companies file to the US Securities and Exchange Commission each year. As mentioned in the abstract, FinBERT was built using the Masked Language Modeling and Next Sentence Prediction tasks. While it was able to achieve commendable results for these tasks, we are interested in making this pre-trained model perform a different task: Sequence Classification. By applying this model to a related, but still different dataset as what it was originally trained on, we're able to explore what the practical implications of a pre-trained model like this is able to provide. For our case specifically, we are training the model on 10-Q reports and predicting the investor sentiment as reflected in the stock price (according to the efficient market theory) in the days immediately following the filing of the report. We train both the FinBERT model and the traditional BERT model (Toutanova, 2018) in this fashion to explore if the domain focused model is able to outperform the

Model	Corpora
FinBERT Prime	10-K (2017, 2018, 2019)
FinBERT Pre2K	10-K (1998, 1999)
FinBERT Combo	All 10-Ks Above

Table 1: Models and their Corpora

Model	MLM	NSP
FinBERT Prime	77.52%	94.50%
FinBERT Pre2K	70.33%	93.00%
FinBERT Combo	75.33%	94.38%
BERT	51.18%	60.88%

Table 2: Model Test Results on 10-Q Filings

general domain model.

2 Background and Related Work

Going into some more necessary detail on FinBERT, the model was trained and tested in a few different iterations. The authors created a corpora that they called SEC2019, which they created by obtaining 10-K filings for 4,392 companies from the years 2017, 2018, and 2019. Additionally, they also created a separate corpora called SEC1999 which consisted of 10-K filings for 4,786 companies from the years 1998 and 1999. By gathering documents from such different time periods, they were able to capture the variation that could've existed as financial language evolved throughout the two decades, while still capturing the meaning of evergreen terms. Finally, they created a third dataset that was a combination of SEC2019 and SEC1999 which consisted of the 10-K filings for 7,272 companies. In doing so, they trained three different models called FinBERT Prime, FinBERT Pre2K, and FinBERT Combo, respectively. See Table 1.

The authors took slightly different approaches in training the three models, training some from scratch and other from the latest BERT checkpoint, taking advantage of the transfer learning capabilities. After playing with a variety of hyperparameters and training the models, they performed Masked Language Modeling and Next Sentence Prediction tasks on 10-Q filings, which are related to the 10-K filings training dataset but obviously different. Upon evaluating, the accuracy scores achieved by the FinBERT models are significantly higher than what traditional BERT (without fine-tuning) is able to achieve. See Table 2.

3 Our Approach

Our goal is to apply this improved model that understands business and finance domain better than traditional BERT to a practical and concrete classification task. Publicly traded companies offer us a good source of data for this task as all of them are required by the Securities and Exchange Commission to file a 10-Q report each calendar quarter where the contents of the report contain a large amount of text commenting on and explaining how the company has performed in the past quarter. Investors would then read these reports and develop a sense of confidence (or lack thereof) in the company, and in turn invest more into the company's stock or pull out their current investment, thus driving the stock price on the market up or down. We're able to use the 10-Q reports as our text feature set and use the stock price fluctuation as our target labels (more on this in Section 4.1 and Section 4.2).

On the model side, we take the traditional BERT base model with twelve encoder layers and build an additional classification layer on top of it (with a dropout). FinBERT was originally fine-tuned off of BERT base as well, and so it also inherently has twelve encoder layers, which we then build an additional classification layer on top of. This gives us a good experimental setup as there is virtually no architectural difference between the models and the only thing that we're testing is the different weights saved in each model (how differently each model "understands" financial language). In both cases, we unfreeze all of the layers and allow them the ability to adjust all weights as each model sees fit. Since we're dealing with a classification task, we are using the Cross Entropy Loss function in evaluating the model.

Once trained, we will be looking at the precision, recall, F1, and overall accuracy scores for each model evaluated on a holdout test dataset. We've opted for these traditional accuracy scores with interpretability in mind. Given the domain of this project, we anticipate people with a finance background (and not necessarily a statistical background) to be a major audience. These traditional accuracy scores mentioned, as well as terms like "False Positives", "True Negatives", etc. are easily understandable, as opposed to others which might require a deeper statistical knowledge base to adequately grasp (e.g. AUC).

4 Experimental Setup

With the capabilities offered in Google Colab Pro, we use a High-RAM session and a GPU Hardware Accelerator to obtain, clean, and engineer our dataset. We then train our models on a variety of different labels that we've created, and we do so in the same way for both models (i.e. number of epochs, learning rate, etc.). As alluded to above, we train the models in the same fashion so that we're isolating the effect of the model weights as much as possible, before measure the performance of each model on the holdout test dataset.

4.1 Dataset

We've acquired our dataset using the SEC's Electronic Data Gathering, Analysis, and Retrieval system (EDGAR) (Securities and Comissions). **EDGAR** houses all official financial documents that companies are required to submit (including 10-Ks and 10-Qs) and allows the public free access. We obtained a list of publicly traded companies that have been actively filing reports from the year 2015 to the year 2020. Utilizing the web-scraping capabilities of **BeautifulSoup** (Richardson), we scraped the 10-Q reports for these companies and saved them as raw text files. Furthermore, 10-Q reports are usually close to 100 pages long and contain massive amounts of text in comparison to what a BERT model is able to make use of. For this reason, we ended up only saving the section of each 10-Q called *Management's Discussion and Analysis of Financial Condition and Results of Operations* or MDA. The **MDA** section is the first block of commentary provided by the company in each report and usually serves as the summary for the entire 10-Q report. This is also what the majority of investors spend their time reading. By using a whole slew of REGEX patterns, we were able to extract just the MDA section for each 10-Q. These sections often look something like this:

> *Total net revenue for the three months increased by $1.8 billion or 140% . . . we benefited from the growth in payment volume processed by existing sellers.*

The MDA section itself, however, is also considerably longer than what a BERT model can take as an input, often spanning a dozen pages or more. Because of this, we first removed all numerical texts from the MDA section (dollar amounts, percentages, dates, etc.) before tokenizing. This decreased

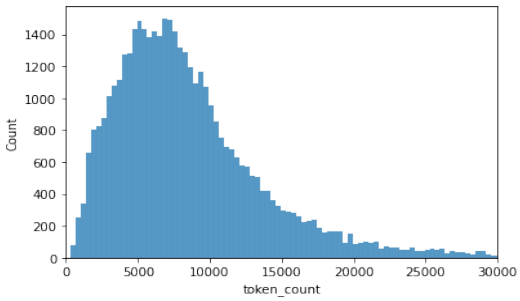

Figure 1: Token Count Distribution of MDA

the token count for each data point since a huge proportion of MDA sections are actually tables with numbers reported in them. By doing so, we also bring to the forefront the actual words of each section that contains crucial meaning. After cleaning the data in this manner, the average MDA section contains about 7,500 tokens. The majority of the 10-Q MDAs contain somewhere between 5,000 to 7,000 tokens. See Figure 18 for the token counts.

Still, we needed to truncate each MDA further to get each one to have a maximum sequence length of 510 (512 including the beginning of sentence token and the end of sentence token). We tried truncating each MDA to just the first 510 tokens, as well as truncating each MDA to only the last 510 tokens. The major issue we discovered with these two methods is that the beginning and end of each MDA contains a lot of disclaimer language (see example below) that does not actually reflect any signals in the commentary:

> *You should read the following discussion and analysis in conjunction with the information set forth within the condensed consolidated . . .*

With this in mind, we ended up truncating each MDA so that only the *middle* 510 tokens remain. With this method, we were able to capture a much higher proportion of meaningful text than either of the other two methods. For the sake of consistency, this was all done using the `bert-base-uncased` tokenizer, the same one that the original authors of FinBERT used. Ultimately, we had a dataset totaling **43,400** rows of MDA sections that we scraped and parsed.

4.2 Target Labels

As for the target labels for our dataset, we used `yfinance` (ranaroussi), a python package that offers a reliable way to obtain historical stock price

data from Yahoo! Finance. With the date that each 10-Q was filed on already in our dataset, we pulled the stock price data for the respective company. The three data points we used from this pull was the closing price on the day of the 10-Q filing, the opening price on the immediate next business day, and the opening price seven days after the filing (or the closest business day). Using these three prices, we engineered our target label classes.

As there is a lot of flexibility in how we're able to define classes using these data points, our goal was to engineer classes that made practical sense while not posing a huge class imbalance issue. For example, one labeling definition was the percentage increase/decrease between the opening price on the next day and the closing price on the same day of the 10-Q filing. This makes practical sense because 10-Q reports are often read by some very intentional investors as soon as they're released and investment decisions are made during market-close hours, meaning that the opening price on the very next day will reflect said decisions. Defining a 5% increase or more as the positive class, and anything below that (including a decrease) as the negative class, we see that there is a very large class imbalance issue as 94% of all data points were in the negative class while only 6% were in the positive class. See Figure 2.

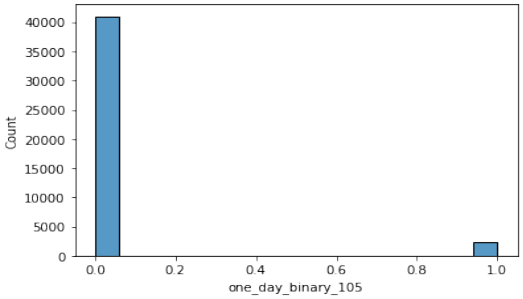

Figure 2: Class Distribution Using Next Day Binary 5%

With this in mind, we ultimately decided to test and use two different target label definitions, one as a binary classification, and another as a three-category classification.

1. **7th Day Binary Increase**: (Figure 3)
   - **Positive Class**: 7th Day Opening Price *greater than* Same Day Closing Price (by more than 0%)
   - **Negative Class**: 7th Day Opening Price *less than* Same Day Closing Price (by more than 0%)
2. **7th Day Three Class $\pm$2%** : $(Figure\ 4)$
   - **Positive Class**: 7th Day Opening Price *greater than* Same Day Closing Price by more than 2%
   - **Neutral Class**: 7th Day Opening Price *greater or less than* Same Day Closing Price by less than 2%
   - **Negative Class**: 7th Day Opening Price *less than* Same Day Closing Price by more than 2%

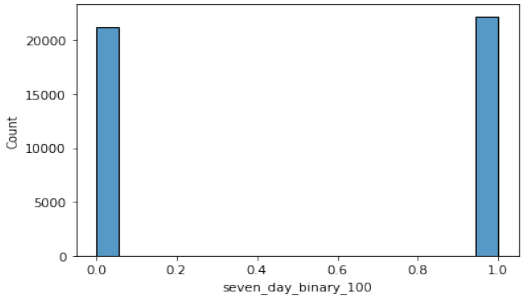

Figure 3: Class Distribution Using 7th Day Binary 0%

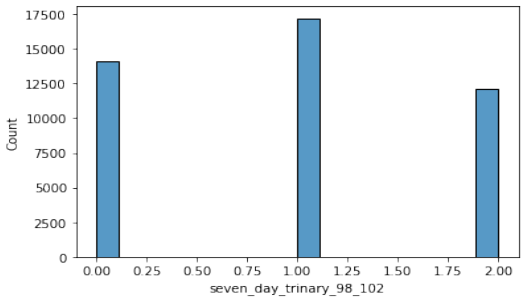

Figure 4: Class Distribution Using 7th Day Three Class $\pm$2%

Seven day period is large enough for other macro and micro-economic factors as well as company related news or events not included in the 10-Q filing to impact the investor sentiment. Here, we are making simplified assumption that no such events take place or even if they do then they have minimal to no impact of stock price. The stock price variation in that case is mainly according to 10-Q filing information, especially based on MDA. There is one more simplified assumption with the labels that the markets are behaving as per Efficient Market Hypothesis which states that the stock prices reflect all available information all the time. In practical conditions, these assumptions are highly likely to fail or disputed. We are not trying to predict stock price with our experiments as it is a futile exercise. Our primary intention is to attempt to see how effective BERT and FinBERT are for sequence classification when they are fine tuned on language used in finance domain. With our simplified assumptions,

seven day period has a slightly different and in a way practical implication compared to the Next Day period. Seven days is enough time for the entirety of the general public and retail investors to read the 10-Q report and make investment decisions in the company (as opposed to only the professional investors). Additionally, we see that the classes are relatively balanced using these definitions, fulfilling our second criteria mentioned above. More examples of target labels we created can be found in the Appendix A.

4.3 Training and Hyperparameters

In training our models, we're making sure that we are fine tuning the FinBERT model in the same fashion as how we would the traditional BERT model. As mentioned above, we are ensuring this happens so that we can compare the potential effect of the differing model weights between FinBERT and BERT. With that being said, we used a PyTorch implementation with the GPUs provided by Google Colab Pro and trained many iterations of each model, changing different hyperparameters, before settling on an implementation that took advantage of our available resources and provided sensible results.

We trained our models on batches of 12 samples at a time. This was mainly determined by memory constraints that exist with Google Colab GPUs that are operating on data points of our size. With a training set of about 32,000 rows, it took approximately 2,700 batches to train on one full epoch. The learning rate for the first three full epochs was set to 5e-6. Further epochs at this learning rate did not see the loss continue to decrease. Because of this, we continued training our models an additional seven epochs at a learning rate of 5e-8. This was a much more appropriate learning rate as the loss continued to decrease as it finished training on a total of ten epochs.

4.4 Results

The resulting predictions from our models have varying degrees of accuracy depending on the metric. As we look at the results from our **7th Day Three Class** models and ignoring the neutral class, we see that the traditional BERT version was able to achieve an overall accuracy of 54.95%, about 3% higher than what FinBERT was able to do. On the other hand, the F1 score that FinBERT achieved was 50.70%, about 2% higher than what BERT was able to do. We see these mixed results because FinBERT was able to have a higher Recall rate. It seems like FinBERT is slightly more capable at identifying when a 10-Q MDA will be followed by at least a 2% increase in stock price seven days later than BERT is. However, while it accurately predicts more True Positives out of all of its Positive predictions, it also predicts a whole lot more Positives in general, which causes it to also have a higher number of False Positives.

The **7th Day Binary** models, on the other hand, showed a slightly different comparison. Both BERT and FinBERT models achieved virtually the same F1 and Accuracy scores (with a slight edge by traditional BERT). In this Binary case, BERT was actually able to achieve a higher Recall rate of 57.72% while FinBERT only achieved a Recall rate of 51.18%. We believe this is the case due to the meaning of this target label. This model is trained to predict positive classes if the stock price increases by any amount at all, and to predict negative classes if the price decreases by any amount. Stock prices fluctuate all the time, and a minor increase/decrease (even as small as 0.0001%) may simply be due to random market forces. Because of this, the textual features we've extracted from 10-Q reports may not contain sufficient signal for our models to perform at a higher level.

See Table 3 for all model accuracy results.

5 Conclusion and Future Work

In this paper, we analyzed the potential benefits that a fine-tuned BERT model on a specific domain is able to provide when we implement transfer learning and ask it to perform a separate practical task. Additionally, we explored the predictability of stock prices and how much signal we can derive from related textual features. We learned that while a fine-tuned model is able to outperform the vanilla model in certain circumstances and using specific metrics, it won't necessarily outperform it in all applications and ways. Our findings here reflect one aspect in which there is substantial benefit in using a domain specific model, but there are plenty of considerations to take into account in the future as we try to better our understanding.

Perhaps the most impactful implementation that we weren't able to fully see through in our project is implementing a document classification architecture in our model. Since we are only using the maximum sequence length that BERT allows us, we are only taking into account 5% to 10% of the

Model	Accuracy	F1	Precision	Recall
BERT 7th Day Three Class	54.95%	48.54%	49.49%	47.62%
FinBERT 7th Day Three Class	51.86%	50.70%	46.41%	55.87%
Uniform Distribution Prediction	50%	50%	50%	50%
Predict All With Most Frequent Class	Most frequent class is Neutral Class			
BERT 7th Day Binary	52.93%	54.99%	52.50%	57.72%
FinBERT 7th Day Binary	52.24%	53.38%	53.34%	51.18%
Uniform Distribution Prediction	50%	50%	50%	50%
Predict All With Most Frequent Class	50%	N/A	N/A	0%

Table 3: Model Accuracy Results

textual information provided in each MDA. While this should capture a decent amount of sentiment that exists in each 10-Q, we are still missing out on an immense amount of information. Given more time to work on and improve our findings here, our very next step would be to implement the ability to take into account more than just 510 tokens for each data point and turn this from a Sequence Classification task into a Document Classification task.

We attempted fine tuning pretrained T5 (PyTorch implementation) on our dataset. However, our experiments could not complete in time to include results from these experiments. In the future work, we would like to explore this and compare it with results from BERT, FinBERT and other implementations such as Longformer that can take more than 512 input tokens.

Acknowledgments

This project was completed with the support and encouragement of University of California, Berkeley, School of Information instructors Paul Spiegelhalter and Mike Tamir. Computational resources were provided by Google Colab Pro.

References

Kyle Lo Iz Beltagy and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *EMNLP 2019*.

Sungdong Kim Donghyeon Kim Sunkyu Kim Chan Ho So Jinhyuk Lee, Wonjin Yoon and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Oxford*.

Rajesh Ranganath Kexin Huang, Jaan Altosaar. 2020. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *ACL Anthology*.

Jieh-Sheng Lee and Jieh Hsiang. 2019. Patentbert: Patent classification with fine-tuning a pre-trained bert model. *EMNLP 2019*.

ranaroussi. yfinance.

Leonard Richardson. Knuth: Computers and typesetting.

US Securities and Exchange Comissions. About edgar.

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Kevin Hanna Vinicio DeSola1 and Pri Nonis. 2019. Finbert: pre-trained model on sec filings for financial natural language tasks.

A Appendix

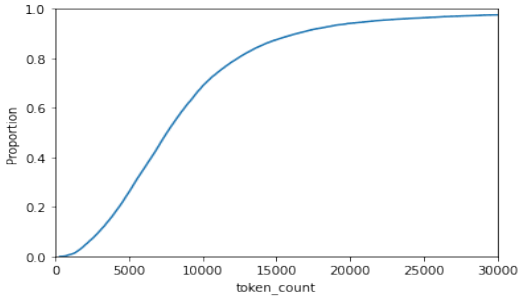

Figure 5: ECDF Token Count

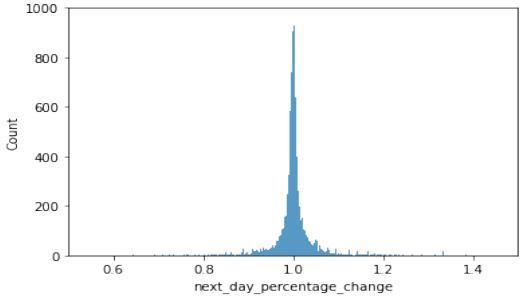

Figure 6: Next Day Percentage Change Distribution

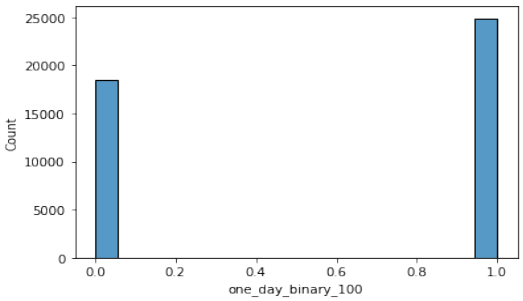

Figure 7: Next Day Binary 0%

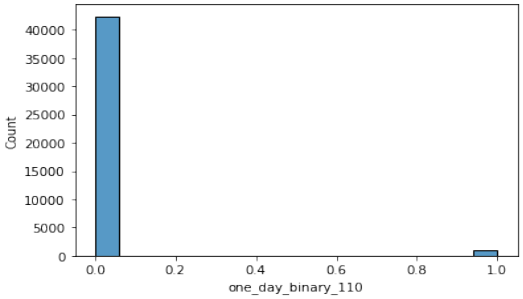

Figure 8: Next Day Binary 10%

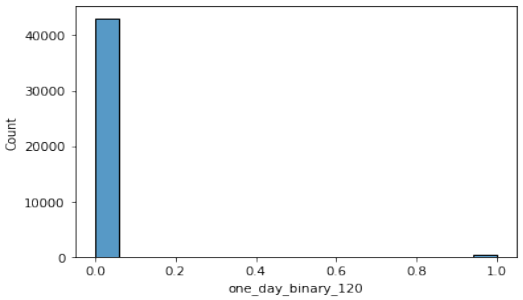

Figure 9: Next Day Binary 20%

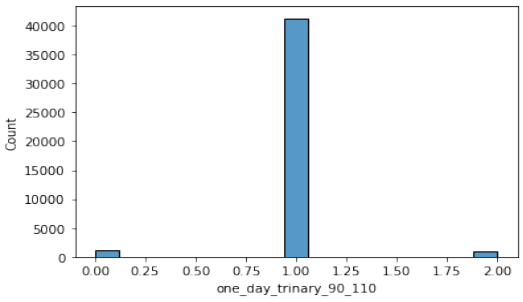

Figure 10: Next Day Three Class 10%

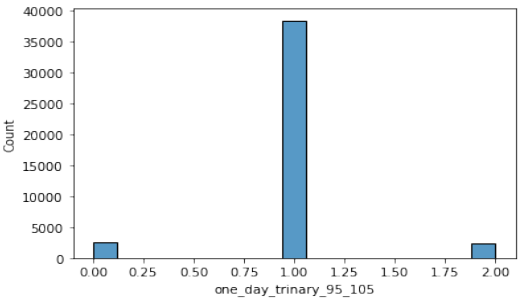

Figure 11: Next Day Three Class 5%

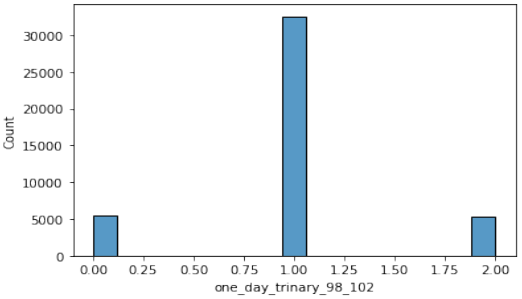

Figure 12: Next Day Three Class 2%

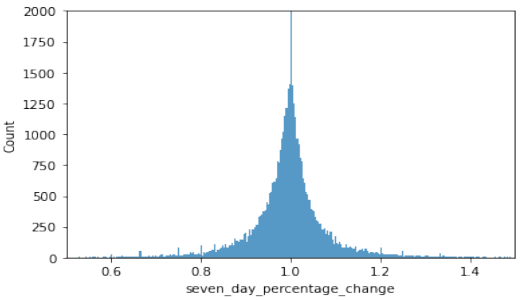

Figure 13: Seven Day Percentage Change Distribution

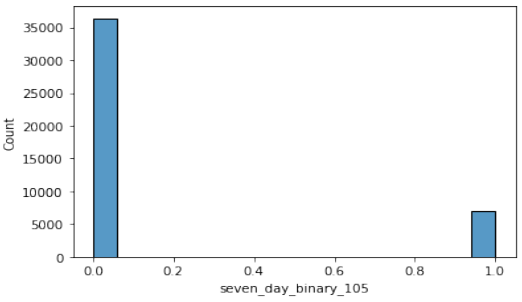

Figure 14: Seven Day Binary 5%

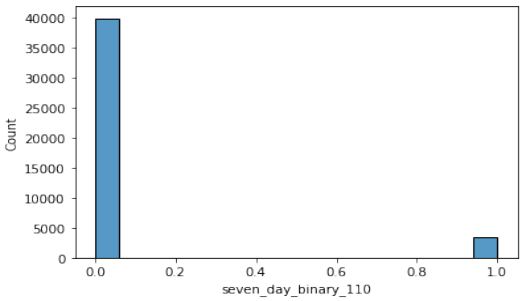

Figure 15: Seven Day Binary 10%

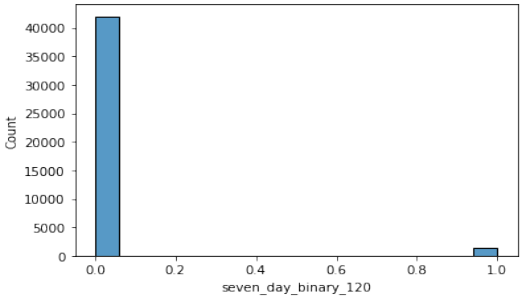

Figure 16: Seven Day Binary 20%

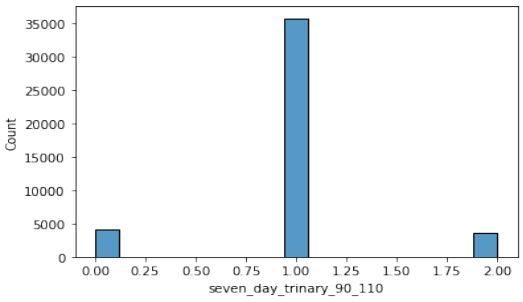

Figure 17: Seven Day Three Class 10%

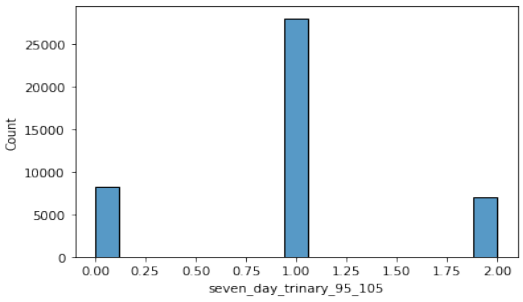

Figure 18: Seven Day Three Class 5%