Predicting Stock Market Trends from News Headlines

Author: Tom Molyneux

April 15, 2024

Supervisor: Dr Neil Mac Parthaláin

This report was submitted as partial fulfilment of a BSc degree in Computer Science and Physics (GF43)

Contents

1 Abstract

With 90% of investors losing money to the stock market, it is evident that humans face significant challenges in interpreting large amounts of data and extracting meaningful insights from their inherent structures. However, this issue is not as prevalent for Machine Learning models running on modern computers. These models possess well-suited functionality to interpret vast quantities of data and provide accurate insights into the structure and patterns of the data at hand.
The primary objective of this project is to apply sentiment extraction analysis to a collection of news headlines and train multiple machine learning models to make the most accurate predictions regarding stock market movements. To achieve this, we will implement a Bayesian Machine Learning model specifically designed for extracting sentiment metrics from the headlines. Subsequently, through further analysis and the utilization of other auxiliary learning models, we will classify the data. Given a sufficiently large dataset, the models should strive to attain a level of effectiveness in extracting structure that surpasses random selection. Ultimately, this will enable us to gain valuable insights into the structure of the data.

2 Literature Review

Due to the nature of this project, having a foot in both financial and technical developments, there is a great amount of literature available covering similar projects. The majority of projects seemed to either focus on Sentiment Extraction or Stock Metric Classification, so combining both provides a fresh project, with novel challenges to overcome.
As someone with an interest in both the financial sector and the Machine Learning capabilities of modern day computing, the concept of this project and the challenges that entail are something I find extremely motivating.

2.1 Similar Projects

The first place I looked after being assigned this project was the developer of the large Kaggle dataset [1] that was suggested by my supervisor Neil ,In our first meeting. This first dataset was an 8 year log of the Dow Jones Industrial Average, gathered between 2008-08-08 to 2016-07-01, providing an excellent amount of data to be analysed. In on-line discussions AARON7SUN the creator notes that he gathered the data via an API linked to the Dow Jones Reddit News page, and although people have since improved upon this collection, this will be an adequate amount to begin working with. Considering the nature of the data collection, it could be supposed that the data has had opportunity to be perverted in some way(or be biased due to the unmoderated nature of Reddit), however in the context of this project this is not a priority and may only need be considered if the final product is released onto "live" data where the bias limits the classifier performance.

The structure of this dataset provides 1989 data indexes with 25 news headlines per index, providing a total of 49,725 data points pre feature extraction. This immediately gives a substantial basis for training classifiers, hence why I have taken the decision after review to use this data over the alternatives, which provided less information with less relevance.

In regards to reviewing other sources, It is important to give an honourable mention to another Kaggle creator [2], not for the database of news headlines, which were not chosen due to the size and quality, but are of interest due to the use of the NLTK library which pointed me toward the first steps of sentiment extraction. This repository by Sid Arcidiacono and Pritam Das uses a smaller dataset but has a very similar objective in the sense of extracting sentiment from news headlines. Where their project succeeded in exploring sentiment extraction in great detail, the lack of dimensions and the restrictiveness in data made the dataset they used inferior to the first choice.

On the contrary, the Kaggle dataset by Khushi Pitroda [3] provides a very sophisticated look into stock market prediction with a well developed dataset scraped from www.nasdaq.com. However, the nature of the data did not contain news headlines but instead stock market headlines(numerical data) and there was no further work into machine model prediction. This dataset was simply reviewed for the depth it went into the stock market metrics and insight it provided into how other people have implemented it into a machine learning model[4].

2.2 Sentiment Extraction Solutions

Following a previous lead to the NLTK (Natural Language Tool Kit) my first review of a language process package was the NLTK documentation found on-line [5]. Claiming to be the "leading platform for building Python programs to work with human language data" immediately offered an array of potentially useful services. My main interest was a set of modules developed in order to extract sentiment from sentences, as this would lay the framework for collating metrics to train on later. With the modules they provide covering most if not all of what I was prospectively to do, the next step was to find examples of the NLTK libraries being used, and evaluate how suitable they are to be included.
The first instance of the NLTK library I found being used to extract sentiment analysis from headlines was a Github repository[6] made by a company called TheCodex. About a project very similar in nature to my own. The company is a computer science teaching resource, and to my own benefit they go into the process of sentiment extraction analysis and buidling a machine learning model in great detail. This can be found on a video resource they published at the link in the reference. The drawback I encountered with this source, and the reason it is not a golden bullet to my project is the lack of depth in both sentiment extraction and classifier learning. It did however provide an excellent example of the use of the NLTK Vader package, which allowed them to extract a number of metrics from news headlines. These included Positive, Negative, Neutral, and Compound metrics which naturally are imperative values.
From the sources that were considered during the Literature Review, the majority of them were based on NLTK sentiment extraction models, which can be found in the following references[7][8].
Although nearly all online resources seem to use NLTK as the industry standard, an article on the Kaggle website[9] outlined the benefits of using the alternative spaCy. Despite this alternative using "better algorithms" than NLTK, it is outperformed in sentence tokenization, and since this will be an important part of the project this consideration can be dismissed.
This essentially summed up the majority of all resources that I found and reviewed in regards to sentiment analysis. Either only NLTK was mentioned

or an alternative was tested but performed sub par in comparison.

2.3 Machine Learner Modelling

The final section the literature review is the choice of machine learning tools. With such a wide range of libraries available, I looked to the similar projects referenced above for an indication of where to look. From the previous projects three libraries had been mentioned multiple times in high regard PyTorch, TensorFlow and Sci-Kit learn.

Having already used TensorFlow in previous projects, this was the first option I reviewed. From previous experience with TensorFlow it is an extremely powerful tool with great scalability and flexibility . The teaching resource GeeksForGeeks.com published a valuable article [10] displaying the capabilities of the library, training a machine learning model on a live apple dataset, and demonstrating the amazing performance. The drawbacks to the TensorFlow model however after reviewing the documentation [11] and examples of usage [12], are that for the precise technical control on offer there is a lot of irrelevant nuances that are required to use the libraries. Since we are wanting to test multiple machine learning models, for the time taken to construct the TensorFlow models, there are quicker and better performing libraries available.

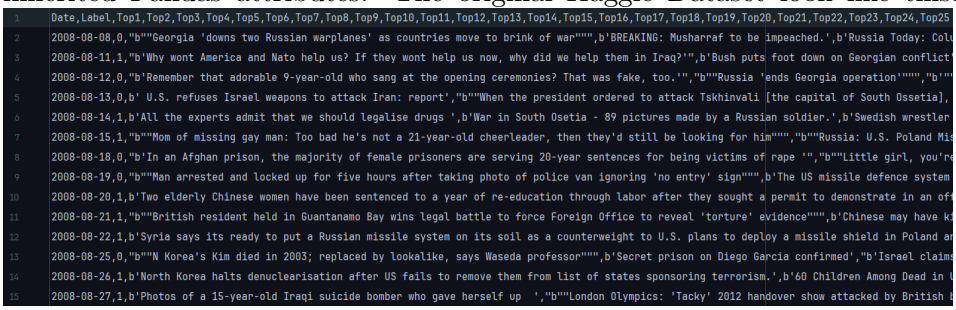
PyTorch is a solid second choice for implementing machine learning to predict stock market trends due to its versatility and robust feature set. Its ease of use and Pythonic syntax make it straightforward to experiment with various deep learning models and algorithms, crucial for developing accurate stock market prediction models. That being said there are certain drawbacks that make this choice less suitable for this project in comparison to the final library. An article by AnalyrtixLabs.com [**?**] compared PyTorch to TensorFlow, deciding PyTorch was better due to it being easier to learn, faster to process and having a wider range of models available. Both were deemed excellent candidates for research and development products.

From the majority of sources provided prior, sci-kit learn was recommended as the library of choice. This is due to it's inherent compatibility with other pythonic libraries such as NumPy, SciPy, Matplotlib, and pandas, all libraries that would be extremely useful to use. Compared to the other machine learning algorithms covered, Sci-Kit learn is an incredibly intuitive library, offering a much larger array of services, and operating at an efficiency comparable to PyTorch. Sci-Kit does not operate using hardware boosting techniques unlike TensorFlow. For these reasons and the abundant example projects to learn from, Sci-Kit seemed the best option out of the three main libraries I have reviewed.

3 Pipeline Design

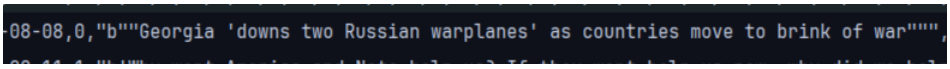
3.1 Data Processing

The aforementioned Kaggle Dataset provides a large quantity of data in the form of a csv file. This file contains the date, 25 headlines from different sources, and a label to indicate the movement of the stock market. This will be immediately formatted into a Pandas dataframe, with the purpose of being able to clean the data in such away that allows us to undergo effective sentiment extraction later on. This process will ensure that the file has been formatted correctly, Syntax inside the headlines has not confused Pandas, and ultimately we can address the data in a useful way via the inherited Pandas attributes. The original Kaggle Dataset look like this:



```
Date,Label,Top1,Top2,Top3,Top4,Top5,Top6,Top7,Top8,Top9,Top10,Top11,Top12,Top13,Top14,Top15,Top16,Top17,Top18,Top19,Top20,Top21,Top22,Top23,Top24,Top25
2008-08-08,0,"b""Georgia 'downs two Russian warplanes' as countries move to brink of war""",b'BREAKING: Musharraf to be impeached.',b'Russia Today: Colu
2008-08-11,1,"b'Why wont America and Nato help us? If they wont help us now, why did we help them in Iraq?'",b'Bush puts foot down on Georgian conflict'
2008-08-12,0,"b'Remember that adorable 9-year-old who sang at the opening ceremonies? That was fake, too.'","b""Russia 'ends Georgia operation'""",b'""
2008-08-13,0,b' U.S. refuses Israel weapons to attack Iran: report',"b""When the president ordered to attack Tskhinvali [the capital of South Ossetia],
2008-08-14,1,b'All the experts admit that we should legalise drugs ',b'War in South Osetia - 89 pictures made by a Russian soldier.',b'Swedish wrestler
2008-08-15,1,"b""Mom of missing gay man: Too bad he's not a 21-year-old cheerleader, then they'd still be looking for him""","b""Russia: U.S. Poland Mis
2008-08-18,0,"b'In an Afghan prison, the majority of female prisoners are serving 20-year sentences for being victims of rape '","b""Little girl, you're
2008-08-19,0,"b""Man arrested and locked up for five hours after taking photo of police van ignoring 'no entry' sign""",b'The US missile defence system
2008-08-20,1,b'Two elderly Chinese women have been sentenced to a year of re-education through labor after they sought a permit to demonstrate in an off
2008-08-21,1,"b""British resident held in Guantanamo Bay wins legal battle to force Foreign Office to reveal 'torture' evidence""",b'Chinese may have ki
2008-08-22,1,b'Syria says its ready to put a Russian missile system on its soil as a counterweight to U.S. plans to deploy a missile shield in Poland an
2008-08-25,0,"b""N Korea's Kim died in 2003; replaced by lookalike, says Waseda professor""",b'Secret prison on Diego Garcia confirmed',"b'Israel claims
2008-08-26,1,b'North Korea halts denuclearisation after US fails to remove them from list of states sponsoring terrorism.',b'60 Children Among Dead in U
2008-08-27,1,b'Photos of a 15-year-old Iraqi suicide bomber who gave herself up  ',"b""London Olympics: 'Tacky' 2012 handover show attacked by British b
```

From here it a little difficult to visualise what we need to change with regards to formatting but the main items to handle are headlines with speech marks, commas, or alternative formatting, as shown bellow;



```
-08-08,0,"b""Georgia 'downs two Russian warplanes' as countries move to brink of war""",
```

Demonstrates a news headline with multiple sets of speech marks. The effect of leaving this in would be a corruption in the punctuation metric, as reviewing the original dataset reveals that this is a product of poor formatting. This is due to inconsistencies with news sources using single and double speech marks, which are formatted differently when converted to csv. Removing this will improve the purity of the data and in turn the final classifiers.

```
,"b'Why wont America and Nato help us? If they wont help us now, why did we help them in Iraq?'"
```

Illustrates how we must ensure that commas inside headlines do not get read as a new element inside the csv formatting. Unchecked this could lead to a problems iterating through the document and cause breakdowns in the code.

3.2 Sentiment Extraction

Following the cleaning of the data, each headline in the dataframe will be passed into a sentiment analysis function of the NLTK library that will allow us to gain valuable metrics about it. Imperative attributes such as the positive score, negative score, neutral score, and overall score will be collected for later use. Accompanying these crucial scores will be a number of auxiliary scores that will be gathered by manual means such as, Word count, Length, punctuation count, n-gram count. All of which have been selected due to the fact they may possibly give a better understanding of the structure of data for the classifiers. As many of these metrics have been chosen due to the nature of machine models performing better with more data. From the 49,725 data points provided in the original, each can have 8 metrics about their indication to the news, scaling our training data to 397,800.

After these metrics have been established, the new compilation of data will construct a new Pandas dataframe in preparation for interfacing with Sci-Kit learn. This dataframe will consist of the 8 metrics taken from the headlines as features, with 1989 indexes.

3.3 Machine Learner Modelling

After the Sentiment Extraction phase of the project we will be working with a second Pandas dataframe with all the data we collected in the previous stage. Iterating over this dataframe we can pass values into Sci-Kit learns various classifiers. The classifiers I have chosen consist of a benchmark Naive Bayes classifier, a Random Forest classifier, and a Support Vector Machine with a radial basis function.

I chose a Naive Bayes classifier due to its simplicity, speed, and capability to handle the high-dimensional data prevalent in this analysis. Despite the assumption of feature independence, the classifier provides reasonable predictions and offers interpretability, allowing me to understand the key factors influencing stock movements. The ease of implementation, computational efficiency, and ability to handle large datasets make the Naive Bayes classifier a suitable choice for my stock market prediction endeavor. Sci-Kit has a pre built Naive Bayes classifier that will allow me to give it the entirety of the dataset to train its mathematical model on, and also a k-fold method for validation afterwards.

The second classifier I decided to use for analysis is a random forest classifier due to its ensemble nature, which combines multiple decision trees to make predictions. This algorithm is effective in handling high-dimensional data, reduces overfitting by random selection of features for each tree, and aggregates predictions for accurate results. Their versatility and success in

various domains, including stock market prediction, make them a valuable choice for modelling complex relationships between sentiment metrics and stock movements.

An SVM with a radial basis function (RBF) kernel stands out as a top choice. This model excels at unravelling intricate relationships within complex data, making it ideal for learning the structure of stock market movements. By leveraging the RBF kernel's capacity to transform data into a higher-dimensional space, the SVM can adequately learn the non-linear nature of market trends.Achieving optimal results with this model hinges on meticulous calibration of parameters like gamma and C, ensuring a finely tuned balance between overfitting and underfitting. Ultimately, the SVM with RBF kernel proves invaluable in for stock market analysis, offering a robust framework for predicting market dynamics.

XGBoost was recommended[7] for its predictive accuracy and ability to handle intricate relationships between datum. By employing a gradient boosting framework to combine weak models into a robust ensemble, XGBoost excels at delivering highly accurate predictions for diverse applications. Its feature importance analysis sheds light on crucial variables, aiding in pattern recognition and decision-making. With built-in regularization techniques like L1 and L2 regularization[15] to prevent overfitting, XGBoost ensures model generalization. As the only classifier that i have not used in previous project, this highly recommended classifier is the least known of the group.

The implementation of all the classifiers will consist of validating the models on a k-fold cross validation technique, with a standard set at 10 folds. This technique will allow for the evaluation of the predictive capabilities of each model chosen [16]. I intend to implement this using a sci-kit learn k-fold function[17] that allows us to run this method on any trained classifiers. Alongside this validation technique I am going to implement another sci-kit method that will provide a classifier report, this will return a number of valuable metrics about the performance of each classifier. These will consist of precision, recall, f1 score, accuracy, and averages. This will provide a basis to evaluate the effectiveness of the classifiers trained.

4 Implementation

4.1 Developmental Choices

I have chosen to implement the design within the PyCharm IDE as it has excellent integration with python modules, and my experience using it makes this a comfortable choice. Further to that I will be using the most recent version of Python at the time, Python 3.12.

4.2 Data Processing

The implementation of the data processing steps were done in concurrency with the sentiment extraction steps, as this allowed for more efficient code, only working when needed. For instance, when converting a string to sentiment metrics, the program will first check that the string is of the correct type. To run an initial loop through the large set of data beforehand would be a great deal more inefficient, hence the approach to clean the data as it is used in the sentiment extraction phase.

Due to the well used nature of the data, there were very little major considerations when cleaning the data. As a result of this the only real implementations were very small handlings of irregular strings. The first few instances of this occurring are during the extraction of n-grams, where each instance of a sentence is cleaned of possible tricky punctuation. Since the handling of this is done concurrently, it has been implemented with a couple of simple if statements. Second to that, during the extraction of the punctuation metric, a metric designed to simply count the number of punctuation characters, the number of speech marks have been handled due to the inconsistencies in news headlines. This has arisen due to a large proportion of news headlines having over 3 sets of speech marks because of a csv translation error, multiplying speech marks where quotation marks should be. The solution for this was not found in a piece of code but the use of excel to identify occurrences across the dataset and remove them using the replace function. This allowed me to clean the data of identified problems.

4.3 Sentiment Extraction

As previously outlined in the design phase of the document, the sentiment extraction section is the first real part of the project where the choices about how to instantiate the project are implemented. To begin with I had decided on using eight intuitive metrics (positive, negative, neutral, compound, ngrams, word count, length, punctuation. During implementation of these metrics it was decided that a further five can be included as complementary metrics. These were: Source, Average word length, Ngrams split into Uni Bi and Tri, and named entity count (Putin, Donald Trump, Russia, Queen Elizabeth). This therefore increased the dimensions of the final data set from 397,800 to 745,875, providing a better basis for training models on.

The implementation of extracting the following values was made very simple through the use of NLTK Vader package. Extracting positive, negative, neutral, and a compound score for each headline was incredibly simple through the SentimentIntensityAnalyzer().polarity scores() method Which when applied to a string of sentence like structure will apply a Naive Bayes sentiment extractor model and calculate accurate scores. These scores will provide us a with a set of values that are intuitively valuable to predicting movements in the stock market. We will review their performance later on.

There were then a number of much simpler auxiliary values that I had decided to include during the design phase. Average word length was a very simple metric to instantiate. Count the number of words using the NLTK tokenizer and divide this by the length of the string. This was chosen due to the insight it gives into if headlines use longer or shorter words during better or worse times for the stock market.

Source was a similarly simple metric to include, and i could perhaps be blamed for overlooking it as a metric during the design phase. This was due to only considering "metrics" in a mathematical scoring sense, however reviewing a few of the similar projects mentioned above, using the source seems to drastically help learners later on. This was again simple enough to implement as the dataset splits each headline by the producer. Although the nature of the source does not give names, like "The Sun" or "The Daily Mail", It provides a number from 1-25 depending on the source, which for training a learning model will work as intended.

The counting of N-grams was implemented very simply as well, simply counting the number of 1, 2 , and 3 character words that appear in a sentence. The only problem solving required during this implementation was the handling of anomalies in punctuation. After running the first iterations of the program I noticed that some headlines included punctuation like "–" or "-" that shouldn't be included in the metrics. Simple if statement ensured that

this was sorted, and the N-gram metric was included.

Finally, I have included an entity metric. This was chosen due to a number of similar projects using an "objectivity score", however since I am already utilising polarity metrics, which act in a very similar way, I chose to adapt this choice and implement entity recognition instead. Entities as described are just tangible things, like a country, person, or building. This should provide a metric in which news headlines can be recognised as having high objectivity or low objectivity. This could be interpreted as an attempt to mitigate news headlines that are in other words "Waffle", or are large but don't say a great deal. This has been implemented using the spaCy package, which allows for intuitive implementation and efficient counting of entities. The $spacy.load('en\_core\_web\_sm')$ download allows the program to understand and classify entities using a "small English pipeline"[19]. Implemented using the help found online [20].

Considering these metrics and their implementation, you may be inclined to think of a number yourself that have not been included. There were quite a few that had been considered during the implementation phase, none more than a time metric which would act as an index. Intuitively this metric would be excellent, providing a scale that allows a learner to understand the development of a stock through time. In other words this could potentially allow a learner to understand that trends may be clustered, following good and bad periods. Due to the nature of the data however, this is not quite possible. If each headline is given an index metric, then 24 other headlines will share the same index and the same outcome. Any learner would quickly learn the structure that identifying one label and one index is the best way to predict future cases. This would create the illusion of an extremely accurate model, but in reality it would be a blunder of design.

4.4 Machine Learner Models

As discussed in the design section, I have chosen to implement four main classifiers (Naive Bayes, Random Forest, SVM with a radial basis function, and XGboost) in order to cover a range of approaches and divine the best classifier. However, during my mid project presentation, post design phase, it was suggested by Dr Richard Jensen that I use an ensemble classifier in conjunction with the ones i have already chosen. The benefit of this being that the classification error is mitigated by other classifier and the scores of each classifier averaged as its own. In the same way that a random forest classifier is an ensemble classifier, but with the same classification model, I have pursued this path by choosing to also implement an ensemble classifier that combines all other classifiers i have chosen to implement. This was accomplished through existing sklearn libraries.

For all classifier models excluding the ensemble classifier, the approach to training and testing data was identical

1. The data is shuffled to remove any structure
2. The data is split 70:30 (training:testing)
3. The selected model is trained
4. K-fold cross validation occurs with 10 folds
5. Classifier performance is evaluated through the output of graphs, confusion matrices, and metrics

Due to the nature of the ensemble classifier, the process is nearly identical with the exception of no.5 (certain visuals and performance metrics) but the process remains largely the same.

4.4.1 Metrics, Matrices, and Visuals

For all non ensemble classifiers, once the model has been trained and 10-fold cross validation has been undergone, the tools that have been chosen to give the best representation and understanding of performance are:

Result Ratio: which gives us the initial split of positive and negative results in the dataset. This is largely important due to the reality that if the data is split 70:30, and I have a model that guesses True 100% of the time, then we will understand why a model with "70% accuracy" is actually an extremely poor classifier. In all cases the data split is around 56:44 (positive:negative) therefore setting a benchmark for any model that has accuracy of above 56% is worth consideration.

K-fold scores and Mean: will return the 10 k-fold validation scores that each section of the model has produced. Alongside this will be the mean

k-fold score, giving a good indication of how accurate the model is as an average.

Standard Deviation and Variance: The standard deviation of a classifier will give us a 1-0 score on how divergent the scores are from the mean. This will provide an intuitive score on how well grouped or precise a classifier may be. As for variance, this score is not a measure of overall accuracy but indicates well over fitting within a classifier. High variance indicates a classifier has learned specific patterns and does not generalise unseen data well. For binary classifiers like the ones trained in this project, variance is not as useful as in other cases, but will still be included for reference.

Confusion Matrix:

Classification report:

References

[1] Kaggle Large Stock Market Dataset (2019) https://www.kaggle.com/datasets/aaron7sun/stocknews By aaron7sun

[2] News Sentiment Analysis for Stock Data by Company (2021) https://www.kaggle.com/datasets/sidarcidiacono/news-sentiment-analysis-for-stock-data-by-company By Sid Arcidiacono, Pritam Das

[3] Stock Market: Historical Data of Top 10 Companies (2023) https://www.kaggle.com/datasets/khushipitroda/stock-market-historical-data-of-top-10-companies By Khushi Pitroda

[4] Stock Market: Historical Data of Top 10 Companies (Community contributions) (2023) https://www.kaggle.com/datasets/khushipitroda/stock-market-historical-data-of-top-10-companies/code By the community

[5] Natural Language Tool Kit website (2024) https://www.nltk.org By Steven Bird and Tom Aarsen

[6] Sentiment Analysis and Visualization of Stock News (2024) https://github.com/TheCodex-Me/Projects/blob/master/Sentiment-Analysis-Stock-News-Final/main.py By TheCodex.com: Avi https://www.youtube.com/watch?v=o-zM8onpQZY

[7] Twitter Sentiment analysis (2022) https://github.com/RobMulla/twitch-stream-projects/blob/main/051-stock-sentiment/stock-sentiment.ipynb By Rob Mulla https://www.youtube.com/live/–rsVYmaHwg?si=-LjZJxtjW5-65GcJ

[8] Stock Market Analysis Using Machine Learning and Python https://everythingcomputerscience.com https://youtu.be/4OlvGGAsj8I?si=6EHZwW8gqTe7l4NA

[9] NLTK vs spaCy (2022) https://www.kaggle.com/discussions/general/299223 By Kartik Guar

[10] Predicting the Stock Market Using TensorFlow (2024) https://www.geeksforgeeks.org/stock-price-prediction-project-using-tensorflow/

[11] TensorFlow Libraries and Extensions (2024) https://www.tensorflow.org/resources/libraries-extensions

[12] Using LSTM's for Stock Market Prediction (2018) https://towardsdatascience.com/using-lstms-for-stock-market-predictions-tensorflow-9e83999d4653

[13] Pytorch vs TensorFlow: Which framework to choose (2023) https://www.analytixlabs.co.in/blog/pytorch-vs-tensorflow/

[14] scikit-learn-vs-tensorflow-vs-pytorch-vs-keras https://ritza.co/articles/scikit-learn-vs-tensorflow-vs-pytorch-vs-keras/

[15] L1, L2 Regularization in XGBoost regression https://albertum.medium.com/l1-l2-regularization-in-xgboost-regression-7b2db08a59e0 By Medium.com: Albert Um

[16] K fold cross validation techniques and its essentials https://www.analyticsvidhya.com/blog/2022/02/k-fold-cross-validation-technique-and-its-essentials/ by Shanthababu Pandian

[17] sklearn model selection kfold https://scikit-learn.org/stable/modules/generated/sklearn.model selection.KFold.html by sklearn documentation

[18] sklearn metrics classification report https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification report.html by sklearn documentation

[19] Trained Models and Pipelines https://spacy.io/models

[20] Named Entity Recognition (NER) in python with Spacy https://www.analyticsvidhya.com/blog/2021/06/nlp-application-named-entity-recognition-ner-in-python-with-spacy/ by Prateek Majumder