Online News Analysis on Cloud Computing Platform for Market Prediction*

Claudia Juarez[1] and Haithem Afli[2]

ADAPT Centre,
Cork Institute of Technology, Cork, Ireland
[1]juarez.moreno@gmail.com,[2]haithem.afli@cit.ie

Abstract. Stock market price fluctuations and predictions have been widely examined; there are two approaches for analysis, fundamental analysis (data from financial records and balance sheets) and technical analysis (focused on past market action). However, past trends cannot predict stock market movement alone; external factors significantly influence them, a notable example being how Twitter comments by Elon Musk affected the stock price for Tesla and the subsequent follow-up by the SEC with the corresponding sanctions. In this highly interconnected society and 24 hours news cycles, we require an extra tool to study the stock market. This research evaluates news articles from financial publications and determines the word patterns that will help make a buy or sell decision, by identifying a combination of words or phrases that indicate if a stock price might go up or down. This research also focuses on the creation of a blueprint for the implementation using cloud technologies to house the financial information, perform the analysis and then present it as a web app for more comfortable use and interpretation. A novice trader could benefit from a simple indicator that provides information on whether the stock might go up or down, as it would facilitate the decision-making process by identifying the exact day for buying or selling, or in case there is no relevant news, then hold the position.

Keywords: Natural Language · Cloud Deployment · Custom Corpora · Deep Learning.

1 Introduction

1.1 Motivation

Stock market analysis is a field that has been studied from several angles, such as economics, finance and statistics, to name a few. Typically for an experimented broker, the breakdown of a stock behavior will be formed by two strategies, the technical analysis, which uses mathematical models to predict the stock variations in the stock charts, and the fundamental strategy.

* Supported by Cork Institute of Technology.

The fundamental analysis comprehends the breakdown of a company's financial statements, historical data, and the interpretation of the summary provided by specialized websites [14]. It also involves plenty of reviews of news, financial articles, and awareness of the company's media presence as new developments or products might bring the stock price up, whereas a news scandal might affect the price and make it go down (albeit temporarily).

The main focus of this research paper and the contribution to the current work is twofold. First, we developed the code for the stock behavior analysis based on the news articles which the end-user will have the ability to select and customize according to their needs.

The second contribution concentrates on cloud architecture development; this paper provides a blueprint for implementing the behavior analysis code in an app that can be accessed over the internet using Python, machine learning, and a commercial cloud platform.

This paper differentiates from past research with these new approaches:

1. It has more extensive data sets, as we created our corpus using financial news articles and stock market opening and closing prices.
2. Our model uses natural language and deep learning as it works best with unstructured data.
3. We provide a practical way for the trader to interact with the predictions through a web app, where they have the ability to choose the source of data collection
4. Simultaneously, the algorithm retrains itself each time the user changes source data with an automated data collection process.

1.2 Contribution

This paper takes away part of the complexity of stock analysis for the average investor, by creating an easy to use web app that will take advantage of machine learning and cloud computing to deliver pertinent information such as a prediction of stock behavior. This tool will help any investor who might not have a financial background but an eagerness to manage their portfolio. It can also help the part-time stock investor to avoid contacting a career advisor and spending a percentage of the profit on these services.

2 Background

2.1 Natural Language Processing Research

Due to this subject's popularity, there has been a myriad of research papers that tested the link between media and stock price movement. Yaojun and Wan have tried to tie stock price fluctuations to social media (although restricted to Chinese social media) and tied it to a 'sentiment analysis,' this means labeling words as positive or negative; even if the result was successful, they only considered Chinese social media accounts that provided financial information, thus having a minimal information source [15]. Nareen found in his study that sentiment directly influences a stock price; however, it did not quantify or offer a prediction [13]. Joshi's study also found a high correlation between price fluctuation and sentiment however lacked a broad set of testing data; they based their research only on historical data from yahoo.com for just one company from news aggregators [8].

Deep learning is widely used with Python as a tool that helps a program learn patterns based on data [4]. This project works based on a correlation between 'words as vectors' natural language processing and sentiment analysis to start mapping keywords/phrases and comparing them to previous stock market historical prices; it also implements word to vector (word2vec) technique to help us train a program in human language nuances such as ambiguities and grammar, while sentiment analysis provides a positive or negative connotation to text and phrases [9].

2.2 Existing Methods

We have taken as starting point three existing research papers and built upon them; Joshi's and Nareen's studies found a correlation between stock movement and sentiment analysis. Yaojun and Wan created a base lexicon that labeled words as positive or negative to relate stock fluctuations to social media posts. We took both ideas and methodologies and created a custom corpora that measured positive and negative in terms of stock market fluctuation and instead of social media news article headlines.

2.3 Cloud Computing

There have been some examples of machine learning models successfully implemented in cloud technologies, although not directly related to stock predictions. There has been evidence that utilizing cloud computing technologies provides a significant reduction of execution time for requests from stakeholders by maximizing the utilization of cloud resources [3].

One such case of analysis is in the healthcare industry. Although it can be applied to any industry, the primary tools Abdelaziz uses are machine learning algorithms deployed in virtual machines. Abdelaziz performed predictions using linear regression and neural networks, finding a considerable improvement in response times of other models using these models in conjunction with cloud capabilities [3].

Currently, there are several tools for analyzing vast quantities of text; one of them is Python NLTK (Natural Language Toolkit). It can help us extract text excerpts and detect patterns, word structure, and frequency to determine meaning and intention[6]. As Feyzkhanov states in his book, serverless deep learning deployment is a novel approach that has the advantages of being scalable, simple, and cheap to start [7]. Most of the academic work does not focus on providing a design for cloud deployment.

2.4 Contribution

We determined that is possible to predict with accuracy the movement of the stock market based solely on news articles taking advantage of cloud computing architecture.

3 Natural Language Processing Implementation

3.1 Methodology

We used model manipulation and local improvement as our heuristic methodology, we changed the nature of the deep learning model to apply it to our particular test case scenario by doing a trial and experiment using different testing data sets; we also created localized improvements by starting with a feasible solution on a working deep learning model and constructing and improving iteratively upon it.[11]

For this research paper, we did not use a standardized sentiment lexicon as we are not looking for a sentiment such as ‘positive’ in the general sense of the word, but ‘positive’ as it is looking for an impact on the stock market. Instead, we created and customized for this particular research necessities’ a custom corpora using different sources and combining them to obtain the data that is relevant for our research. [11]

We analyzed the headers of various news articles from online news outlets based on a stock ticker and supplied a prediction for said stock price movement. The Natural Language Processing or NLP model is housed in cloud-based architecture and made available to the public for ease of use in a web app. We created a custom corpus that worked with the NLP prediction model forgoing using a pre-trained model.

We chose this approach because, upon closer inspection of the individual texts, they contained a lot of ’noise’ such as advertisement, excerpts of other pieces, links, or author’s bios, which were not associated with the article on hand. Headlines of articles are a synthesis of the entire sentiment of the article.

3.2 Data Creation

For the creation of the data set, we used an API to retrieve the publication name, headline, source article and date of publication from Google News and then paired it with the stock movement for that particular day; if the news fell on a weekend or a holiday, we moved to the next business day.

We calculated the percentage of change for that day, either up or down on price, and since these numbers varied widely, we normalized on a scale from 0 to 1, we did this operation for every set of stock ticker and news article.

The main two discoveries we made after creating this custom corpus were that the data set was broad enough to provide information on global markets alongside the search of a single news resource and that the paid API provides more online publications even after just indicating 5 to search through.

3.3 Prediction Models

We selected this model because the Keras/TensorFlow Model can be used for fast prototyping and combined with TensorFlow is optimal for a production-ready system; the downside is that we do not have much control over the code as it is best for fast experimentation. We are working with neural networks that cannot work on raw data text, so the next step is to convert it to tokens, which are just integers. [12]

Tokenize; The tokenizing method works by going through the complete dataset and counting the number of times each word is used and then making a vocabulary where each word gets an index; this way our data samples are converted to numbers called tokens [12]

Pad-Truncate; The sequences are padded/truncated to make sure they are all of a standardized size by taking the average number of words in all the sequences and add two standard deviations to ensure we are aiming at 95% of the data; the padded works by adding zeros to the sequence. [12]

Embedding; Even after converting the words to tokens, a neural network still cannot work on this data due to vocabulary limitations and semantics; so, we use embedding; it works by converting integer tokens into real-valued vectors. This technique is known as representation learning, which is a way to obtain a real-valued representation from a text while preserving the semantics. [12]

Recurrent Neural Network (RNN); The result of the embedding process is a two-dimensional matrix called a tensor, which now is in the correct format to be used in an RNN; its main characteristic is that it can process sequences of arbitrary length. Each layer is dependent on the result of the layer before it, which makes it perfect for natural language processing. [12]

Sequential Model; In this model, we run 3 activation layers plus one dense layer that provides the numerical value. [12]

Activation Function; We use sigmoid as an activation function; it is an excellent generic distribution that handles well randomly occurring events and works best with a small number of layers; also, the sigmoid activation provides us a range of values between 1 and 0. [12]

The result provided is a number between 0 and 1, where closer to zero means that the price of the stock is more likely to go down and closer to one means the price of the stock is more likely to go up.

The Deep Learning model workflow and script we use is based on Menshawy's approach to analyzing move reviews and works the following way:

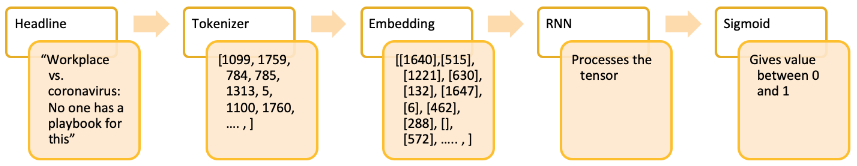

Fig. 1. Deep Learning models Workflow [12]

1. Load the train/test set and preprocess data to:
   (a) Normalize result behavior between 0 and 1
   (b) Fill empty lines with zeros.
   (c) Select the size to train and test sets
   (d) Convert to array
2. Tokenize train/test set data
3. Ensure all sequences have the same length
4. Pad or truncate
5. Embedding or Vector creation
6. RNN model
7. Train model against the test set
8. Evaluate model accuracy
9. Try out with an example.

The deep learning models gives an immediate prediction for the latest news article, it is printed in-screen and saves the result in a CSV file.
Input1: User input of stock name and online publication
Input2: Custom Corpora Data Set
Output1: Print value of model result, between 0 and 1.
Output2: the results of the model are saved in a CSV file.
The CSV file saves all queries for later processing and ingesting into the train/test data set.

These archived results keep incrementing as the app gains popularity and more users query different stocks/publications combinations creating a bigger archiving file.

Model Learning Cycle
The archived results get verified daily through a cron job, archived data gets processed and verified; if enough lines for the same stock (200) are recorded in the file, then said lines would get processed, reusing the script we used for gathering the initial data set. These lines will get their financial information attached to them, the percentage of change calculated, and value normalized. Then the data that has a normalized value of 1 or 0 will become part of the primary test/train data set and flagged 1, which means it was already processed; the workflow is represented on figure 2 Detailed Script Workflow.

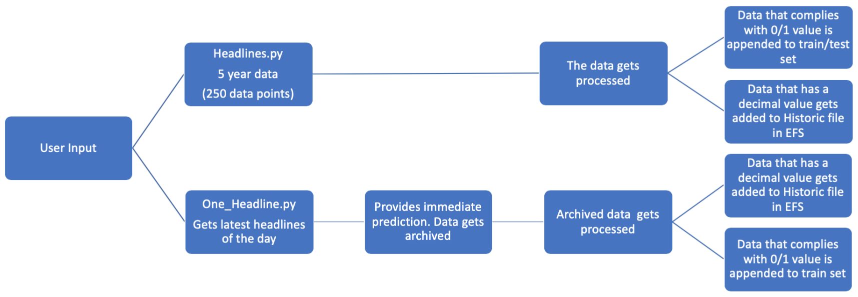
User Input
Headlines.py
5 year data
(250 data points)
The data gets processed
Data that complies with 0/1 value is appended to train/test set
Data that has a decimal value gets added to Historic file in EFS
One_Headline.py
Gets latest headlines of the day
Provides immediate prediction. Data gets archived
Archived data gets processed
Data that has a decimal value gets added to Historic file in EFS
Data that complies with 0/1 value is appended to train set

Fig. 2. Prediction System Architecture

4 Experiments

All scripts were created using Python, Python libraries, and custom APIs; the code for the final scripts is available at the private GitHub repository: https://github.com/claudia0juarez/Thesis

4.1 Parameters

The parameters used in this research were: model accuracy and prediction results, the model accuracy will measure the performance of the model while the prediction results are calculated based on a comparison between the model prediction and the actual stock market movement.

4.2 Web Scrapper Development

The first step in the process was to develop the scripts that will fetch the data from online resources (news articles/web publications). We used Python and the BeautifulSoup library for the web scraper due to the ease of parsing the HTML files. For scanning the web for relevant news, we use a paid API that executes search requests in several search engines called SerpWow. [2]

4.3 Deep Learning Model Fine Tuning

The deep learning model fine-tuning was run using proof of concept tests and a small data sample of 9 news articles with an assessment based on stock price movement; we ran our model and then determined the level of success by comparing the results with the real data.

The deep learning model provides a result in the range of 0 to 1 where:

0 to 0.5 is a negative result meaning the stock is most likely to go down 0.5 to 1 is a positive result meaning the stock is most likely to go up

POC Test1 The first proof of concept test ran with a custom corpus size of 107,000 headlines. The model results remained pretty much in the same range; as shown on table 1 POC Test1 with Custom Data Set in page 8 we are still not getting a reliable prediction.

No	Model Accuracy	Prediction Results
1	69.21	0.49
2	69.31	0.47
3	69.27	0.46
4	69.35	0.47
5	69.22	0.48
6	69.13	0.46
7	69.21	0.47
8	69.33	0.47
9	69.15	0.47
10	69.26	0.48

Table 1. POC Test1 Custom Data Set

No	Model Accuracy	Prediction Results
1	71.8	0.38
2	71.92	0.37
3	71.49	0.35
4	72.08	0.37
5	71.92	0.38
6	71.68	0.38
7	72.31	0.38
8	72.29	0.35
9	71.33	0.38
10	71.66	0.39

Table 2. POC Test2 Custom Data Set

POC Test2 With an extended custom data set of 173,000 data points; again, the accuracy of the model greatly improved as shown on table 2 POC Test2 with Custom Data Set on page 8; however, the results were again in the same range and not providing any valuable feedback.

POC Test3 The custom data set we used has test values ranging from 0 to 1, so for the next test, we only used the data that has naturally 0 (negative) or 1 (positive) to train the model as shown on table 3 for New Testing Data on page 9.

After running the model with this new data set, we had a better accuracy result, and the actual results with the model started to vary from one heading to the next, although it was only correct 3 out of 9 times the main issue was that we had a minimal test/train data set of only 2,500 records as shown on table 4 for New Testing Data on page 9.

Table 3. New Testing Data

No	Stock	Headline	Date	Open	Close	Assessment Based on Stock Movement
1	Microsoft	Workplace vs. coronavirus: No one has a playbook for this	05-Mar-20	166.045	166.27	Positive
2	Microsoft	Pentagon asks to reconsider part of JEDI cloud decision after Amazon protest	12-Mar-20	145.3	139.06	Negative
3	Microsoft	Windows 10 Warning: Anger At Microsoft Rises With Serious New Failure	09-Feb-20	183.58	188.7	Positive
4	Microsoft	How corporate IT is entering the multi-cloud	14-Mar-20	140	135.42	Negative
5	Apple	Microsoft's Massive Stock Gains May Be Far From Over	09-Feb-20	314.18	321.55	Positive
6	Tesla	Tesla's Sales Fell 68% In The Netherlands And 92% In Norway In February	02-Mar-20	711.26	743.62	Positive
7	Starbucks	Starbucks Is Bringing Beyond Meat To Its Canada Locations	26-Feb-20	82.6	80.67	Negative
8	Samsung	Samsung Unveils Samsung Galaxy S20 Series With AI-Powered Camera	14-Mar-20	2209.7	2209.7	Neutral
9	Amazon	Amazon's Stock May Jump Following Quarterly Results Despite Rising Costs	26-Jan-20	1820	1828.34	Positive

Table 4. POC Test3 with Custom Data Set

No	Model Accuracy	Prediction Results	MODEL	ACTUAL STOCK	MODEL VS ACTUAL STOCK
1	75.17	0.61	positive	POSITIVE	🟩
2	76.75	0.31	negative	NEGATIVE	🟩
3	73.51	0.26	negative	POSITIVE	🟥
4	73.57	0.69	positive	NEGATIVE	🟥
5	72.29	0.39	negative	POSITIVE	🟥
6	72.94	0.44	negative	POSITIVE	🟥
7	72.61	0.38	negative	NEGATIVE	🟩
8	76.68	0.47	negative	NEUTRAL	🟥
9	75.19	0.48	negative	POSITIVE	🟥

POC Test4

We did a round of testing converting the entire data set (173,000) and normalizing their score values to either 0 or 1 depending on if it was higher or equal than .5. With this test, we discovered that this model behaves similarly to a Naives Bayer model as it only works with naturally 0 and 1 values.

Table 5. POC Test4 with Custom Data Set

No	Model Accuracy	Prediction Results	MODEL	ACTUAL STOCK	MODEL VS ACTUAL STOCK
1	75.2	0.66	POSITIVE	POSITIVE	🟩
2	72.35	0.69	POSITIVE	NEGATIVE	🟥
3	76.18	0.62	POSITIVE	POSITIVE	🟩
4	77.77	0.6	POSITIVE	NEGATIVE	🟥
5	75.97	0.62	POSITIVE	POSITIVE	🟩
6	77.4	0.6	POSITIVE	POSITIVE	🟩
7	72.15	0.62	POSITIVE	NEGATIVE	🟥
8	77.8	0.65	POSITIVE	NEUTRAL	🟥
9	76.78	0.61	POSITIVE	POSITIVE	🟩

The accuracy remained in the same range; however, the result returned to the same behavior, only giving a value within the same range as shown on table 5 POC Test4 with Custom Data Set on page 10.

The best way to obtain better results with the deep learning model is by using a good data set both in quality and quantity. Quality meaning that only truly 0 or 1 results will provide a good result, there is no use in artificially moving the value set to 0 or 1 as it will throw the overall model. Also, the quantity, with a more significant data set, the results get consistently better.

4.4 Results

After running these POC and refining the model with the custom data set, we ran the custom script to get more data points (from 2500 to 3500) and re-run the test with a more prominent test set (around 272 headers). We ran 3 tests with an incrementally higher number of test sets:

Test1 (2505 test set)

The 272 news headlines had a 46% accuracy when tested along with the custom test set of 2505 data points, which meant that 124 out of 272 headlines were correctly predicted, and the stock price moved accordingly to said value.

Correct Prediction	124	46%
Incorrect Prediction	147	54%

Test2 (3521 test set)

For the second test the 272 news headlines had a 43% accuracy when tested along with the custom test set of 2505 data points, which meant that 116 out of 272 headlines were correctly predicted, and the stock price moved accordingly to said value.

Correct Prediction	116	43%
Incorrect Prediction	155	57%

Test3 (3789 test set)

To get the final set, I needed to create 264,000 overall custom data set and then only use the 0 and 1 values; this test retrieved a 63% of accuracy that represented 170 correct predictions out of 272.

Correct Prediction	170	63%
Incorrect Prediction	101	37%

To get a better prediction result, our work shows that there is a need to keep incrementing the custom test/train data set. However, it is very resource-intensive as it takes about a day to get around 100,000 results (provided the paid APIs don't throw an error), and from this set, we will still have to filter the 0/1 results, which are about 1,000.

5 Cloud Architecture

Service-oriented-architecture or SOA is known for increasing the capability of an enterprise to address new business requirements with minimal cost, resources, and time overheads. We based the development of the cloud-based architecture of our prediction system on this framework.[5]

The blueprint was based in SOA to maximize the cloud benefits; it can leverage cloud computing resources as services contained within itself; it will help layout the design of services that will increase usability and durability as well as the blueprint for design, development, and deployment. [10]

Data to Cloud Roadmap:

1. Define the data. The data was taken from news publications (financial, and well-reputed sources). [5]
2. Define the services. We used AWS, although this layout might be applicable to any other cloud service provider, along with Python libraries, and paid APIs. [5]
3. Define the processes. We used ETL (Extract/Transform/Load). [5] which is a quantitative methodology based on observations and experimentation.
   Extract; Automate the data collection with web scrapping tools for these parameters: renowned online financial publications and historical data on market prices for the past years (opening price, closing price).
   Transform; Use data analytics tools to correlate the price changes of Company Stock vs. news articles mention on the same day. With this information, we created custom corpora and utilize NLP for training a model into determining the sentiment/vector and the effect on the price (high or low).
   Load; Create a cloud-based web app to review different stocks, followed by the creation and documentation for a blueprint design.
4. Define governance; it is the ability to control changes to services and the usage of said services; we must control how our data is accessed, deleted, added, and altered employing processes procedures and technology. We used the AWS in-place security protocols to make sure that our Data is persistent and safe. [5]

5. Define which candidate data, services, and processes should live in the cloud and which should live on-premise (if any). As for this project, all of our data will resides in cloud services. [5]

The high-level architecture of the prediction model, as shown in figure 3 Cloud High-Level Architecture is an integration of three AWS subsystems, each of them housing a subsection of the overall process.

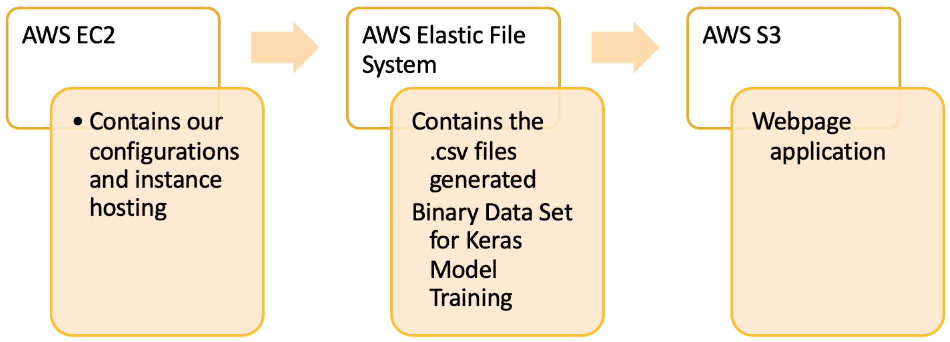
AWS EC2
• Contains our configurations and instance hosting
AWS Elastic File System
Contains the .csv files generated
Binary Data Set for Keras Model Training
AWS S3
Webpage application

Fig. 3. Cloud High-Level Architecture

EC2 hosts the instance, application files, the network security setups, load balancer, the security groups, the VPC and the certificate manager. The EFS (Elastic File System contains the CSV files created by the application and the data set and S3 contains the HTML web application.

The more detailed process described in figure 4 Detailed Cloud Process on page 13 starts with the public internet and the main website www.moneyplease.trade , the request goes through the CloudFront, which warrantees that the edge services of AWS will be close to our primary users; at the moment, we are only using the EU, USA, and Canada as this is the starter level. However, we can upgrade in the future to include more edge zones.

The Cloudfront distribution is connected to an S3 bucket that contains our web app HTML file. We choose S3 because, unlike our CSV files, the content of this bucket will be static. This content is the one distributed to the static edges.

CloudFront relies on the Certificate Manager to generate trust certificates and to have a secure connection; the CM creates and updates the certificates from now on, so we do not need to worry about it.

The communication between the S3 bucket/the load balancer and the EC2 instances is hosted on our virtual private cloud and not over the public internet. S3 will function as our storage system, it works best for hosting our HTML file because it is not primary for reading/writing. Whenever we require to change this file, we will need to download a copy of it, modify it and reload it again as changes in our webpage occur.

S3 will connect then to the load balancer; at this point, it is only managing a single instance. However, it can be set up for escalation (managing recurring instances in case the demand increases)

In EC2 is where the Python scripts reside, they are all managed by a Flask interface that is waiting for the user input to start the process, once the primary process starts it reads/writes from the CSV files in the EFS (elastic file system).

The EFS contains all the working files in CSV format, the data set used for training and testing of the deep learning model, the results from the searches, the historical headlines, the financial information, and the complete result data set. These files will continue to grow each time a user inputs a new search; the EFS will grow with them and will place them in the correct availability order, as files that do not get much use will have a lower priority than for example the binary data set that works with deep learning model during each run.

Another advantage to EFS is that it can be deployed and mounted to all instances and work seamlessly with them if we need escalation.

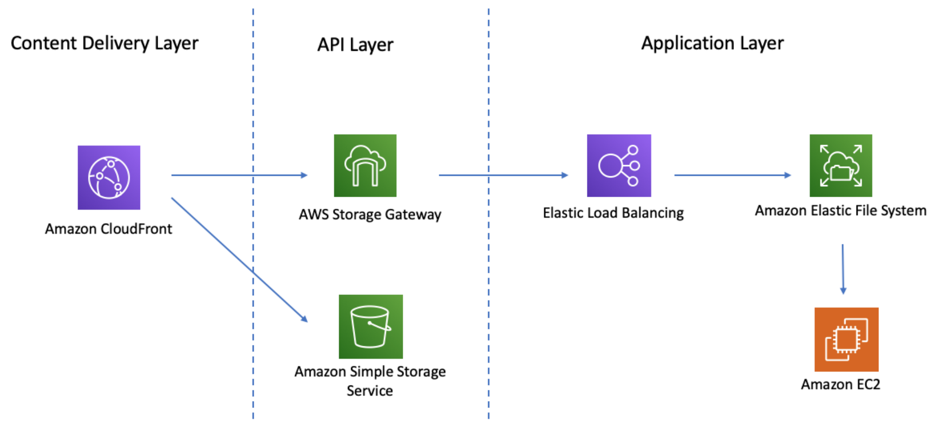

Fig. 4. Detailed Cloud Process [1]

5.1 Integration in web-based application(interface)

Flask is the microserver used to run the application; we selected it due to its integration with Python and its straightforward interface with our already working code; the stack for the backend is shown on figure 5 Backend Stack on page 14

To have a correct division of responsibilities in our app server, we cannot use Flask as the webserver, we needed to use HTML technology; this way we can do front-end developments independently from the back end. Flask has standard methods that allow communication with the front-end; this way we ensure that our deployment is tech agnostic, meaning that we can have future mobile apps or different web applications without changing our main code.

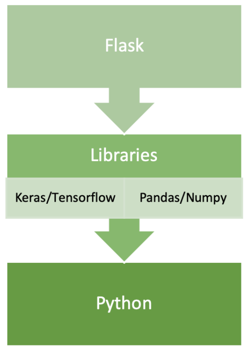

Fig. 5. Backend Stack

For the content distribution, AWS CloudFront is connected to the origin of the data. Then the DNS registries are updated for the domain to work along with CloudFront as shown in figure 6 Backend Distribution.

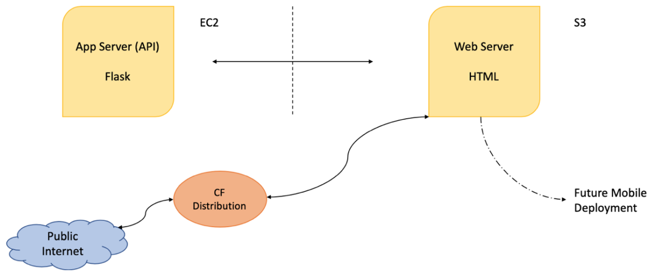

Fig. 6. Backend Distribution

6 Conclusion and Discussion

This work was successful in implementing a prediction model with a 64% accuracy in prediction of stock market price movement; we were also successful in creating a feasible blueprint for a deployment entirely on cloud premises services.

The most important part of the prediction model is the customized corpus made from a suitable binary dataset (only 0 and 1 values after normalizing). If we do not have a dataset containing the parameters we are evaluating, the results are not going to be representative or useful for the type of prediction we are after. A small accurate test set brings better predictions than a big messy one. This corpus will be extended with the new data searches provided by users to retrain our model.

The pay-as-you-go services for the cloud provider and API's are required, as the model grows we will need to pay for the services to keep our implementation going; the model was successfully deployed in the free-tier applications, and it is possible to scale all of the applications once the work extends its size and scope.

For future work and further research we might want to explore the growth of the cloud deployment, at the moment, we are using CSV files in an elastic file system; however, if the data keeps growing as intended this might not be the best use of cloud computing resources, for the next phase in the project we might want to add a relational database and convert the CSV files to a SQL format.

Also, we are not saving any user's personal information or manipulating any other sensitive information; however, if the application grows, we might want to add authentication based on user emails or social media.

Another improvement opportunity worth mentioning is the webscrapper development, we might want to add priorities for the websites that were crawled for the custom corpora; as it stands, the webscrapper takes headlines information for different sources (blogs, smaller news outlets, opinion pieces) as it is focusing on volume for the creation of the testing data set. If this first step gets refined and the corpora not only increases in size but in quality of the headline, it will ideally increase its reliability.

Also the deep learning model used was a popular teaching model, we might consider improving the model type to increase reliability. Although at this point it is only using one computer it is set up for future growth within the cloud space

These considerations for future releases might be excellent additions and nice-to-have, the objective of the research was met with a minimum viable product or MVP that gives a prediction based on a selected publication plus a stock name and is deployed entirely in a cloud environment.

References

1. MicroServices Architectures on AWS.
https://www.slideshare.net/AmazonWebServices/microservices-architectures-on-amazon-web-services

2. SerpWow - Google Search Results API, https://app.serpwow.com/playground
3. Abdelaziz, A., Elhoseny, M., Salama, A.S., Riad, A.M.: A machine learning model for improving healthcare services on cloud computing environment. Measurement: Journal of the International Measurement Confederation (2018). https://doi.org/10.1016/j.measurement.2018.01.022
4. Akshay, K., Shivananda, A.: Natural Language Processing Recipes: Unlocking Text Data with Machine Learning and Deep Learning Using Python. Apress (2019)
5. Banke K., Slama D., K.D.: Enterprise SOA: Service-Oriented Architecture Best Practices. Prentince Hall (2004)
6. Bird, S.: Natural Language Processing with Python. O'Reilly Media (2016)
7. Feyzkhanov, R.: Hands-On Serverless Deep Learning with TensorFlow and AWS Lambda. Packt Publishing (2019)
8. Joshi, K.: Stock trend prediction using news sentiment analysis. Tech. rep. (2013)
9. Lamons, M., Kumar, R., Nagaraja, A.: Python Deep Learning Projects. Packt Publishing (2018)
10. Linthicum, D.S.: Cloud Computing and SOA Convergence in Your Enterprise: A Step-by-Step Guide. Addison-Wesley Professional (2009)
11. Mathirajan, A.I.S.K.N.K.M.: Management Research Methodology: Integration of Principles, Methods and Techniques. Pearson India (2006)
12. Menshawy, A.: Deep Learning by Example. Packt Publishing (2018)
13. Naren, J.: News analytics and dual sentiment analysis for stock market prediction. IEEE International Conference on Big Data Analysis(ICBDA) (December) (2017)
14. Piard, F.: The Lazy Fundamental Analyst: Applying Quantitative Techniques to Fundamental Stock Analysis. Harriman House Ltd, (2014)
15. Yaojun, W., Wan, Y.: Using social media mining technology to assist in price prediction of stock market. IEEE International Conference on Big Data Analysis(ICBDA) (2016). https://doi.org/10.1109/icbda.2016.7509794.