**Daily News Stock Prediction**

Ayush Kapoor

Sannidhya Malpani

***Washington State University***

December 10, 2020

# **Introduction**

**Abstract**

For our Data Science Project, our team had chosen to analyze three datasets which were retrieved from Kaggle (reference is given below) and build a predictive model around it. Our project will involve using the eight years of daily news headlines from the datasets provided, and building a predictive model to forecast the semantic analysis using Deep Learning algorithms and implementations.

We will be taking the data from 2008-08-08 to 2014-12-31 as our Training Set, and compare it with the testing dataset which would be the following two years of data. (2016-2018)

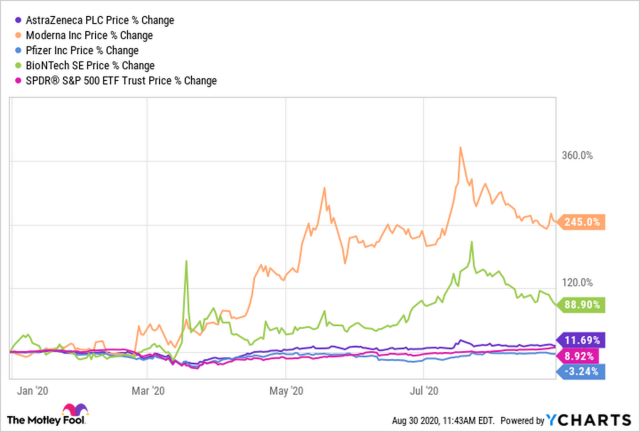
Primarily we used google cloud tools to collaborate. To work on the code we used google colab. To write reports we used google docs. To further share different resources we are using texting platforms such as iMessage and WhatsApp. **The link to our code is provided** [**here**](https://colab.research.google.com/drive/1muSRzTUXFqL1Vd5Ic5_nR5ikQ-kiNm-5?usp=sharing).

**Background**

Analyzing and predicting the movement of the stock market is still considered as increasingly strenuous and back-breaking due to the market being affected by a variety of factors. Some factors we will be indulging into are textual analytics of the daily news headlines, significance of Customer Behaviour and the overall Announcement Effect.

Also known as the “Media Effect”, the announcement effect refers to the instant as well as overtime impact that any news or public announcement has on the financial market. The potential for such an announcement to yield a negative outcome is known as a “**headline risk**”. Such pessimistic news normally causes investors to sell stocks and stretch from political uncertainty to company mergers and acquisitions in the form of either reports,press releases or in this case News Headlines.

“Shares of Moderna are shooting higher today after President Trump announced the company won a contract to produce 100 million doses of its COVID-19 vaccine candidate -- but the ask is not remotely possible, at least not yet.”



*Figure 1: Moderna Stock Fluctuation*

While Moderna blitzed the media, it revealed very little information — and most of what it did disclose were words, not data. That’s important: If you ask scientists to read a journal article, they will scour data tables, not corporate statements. With science, numbers speak much louder than words. Therefore, negative news tends to normally cause individuals to sell stocks. A bad report, any sort of political uncertainty contributes to selling pressure and a decrease to price in stock .

**Objective**

Our objective would be to enhance text mining methods by using more features to represent the textual data and reveal how intense feature selection will reverb classification accuracies.

To be clear, our intention is to look not only on how news affects stocks but quite the opposite too. On one side we are determined to use different classification methods in order to predict whether the news broadcasted that day would fluctuate the price of the DJIA stock. But on the other hand, we also know that algorithmic trading has completely overthrown its surrounding industry, enhancing the primitive approach towards the stock market. Over 70% of all trades conducted in the United States of America are handled by bots.

# **Problem**

## **Dataset Analysis**

In our project, we mainly dealt with three datasets that were provided. There were comma separated value (csv) format and are briefly explained below:

1. News Data

This dataset includes the historical news headlines from Reddit World News Channel, (which is ranked by reddit users’ votes) with a range from June 2008- July 2016. In this file, there are two columns, which are : “dates” and “headlines” All the news is ranked from top to bottom with respect to how ‘hot’ or ‘exclusive’ (Not how accurate). Therefore, there are 25 news columns and a date column.

1. Stock Data

This dataset includes the stock data of Dow Jones Industrial Average (DJIA), which is the stock market index that measures the stock performance of the 30 largest companies that are listed on the stock exchanges in the United State of America.

It is considered as a widely-watched benchmark in the country's blue-chip stocks. (stocks with large market capitalizations). The day to day movements of the Dow Jones Industrial Average are by professional money managers, market analysts and individual investors as these 30 stocks may dictate the entire stock market.

1. Combined Dataset

The third dataset we are dealing with is the Combined Dataset which contains the above mentioned datasets presented neatly in three columns. Starting with the “Date”, then the “Label” followed by the range of “News Headlines” from Top1 to Top25. This is constructed as a Binary classification task. Hence there will be only two types of “Labels”(Labels is a column in the dataset). If the observation is labelled as 1, it implies that the stock rose or stayed the same that day whereas if the observation is labelled as 0, the stock sunk.

* + Wherever the observations are labelled as “1”, it implies that DJIA Adjusted Close value either rose or stayed the same with respect to the ‘Date’ specified.
  + On the contrary, the “0” labelled observations show when the DJIA Adj Close value decreased.

One of the initial findings we needed to overcome were the sentiments that every news brought the dataset. Using the syuzhet package in R-studio, we were able to count the number of words by sentiment. The package comes with four sentiment dictionaries and provides a method for accessing the robust, but computationally expensive, sentiment extraction tool. Furthermore, we extracted the sentiments per day for the entire combined dataframe. The implications and findings are better reasoned out in the results section.

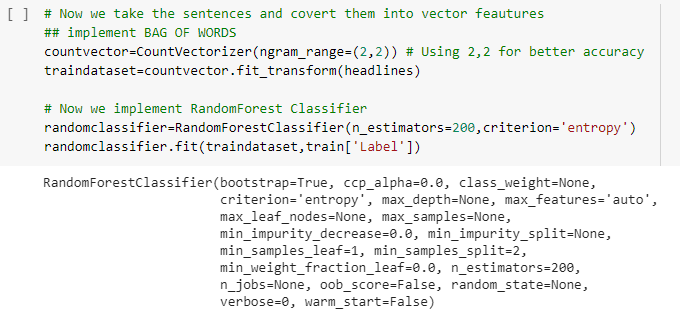
## **Proposed Solution**

### **Naive attempt:**

Our initial attempt was to clean and scout the dataset to reveal what kind of variables and frequencies we were dealing with. After reading in the ‘Combined’ dataset and splitting into training and test data, we replaced the missing values with whitespace and got rid of all the redundant non-word HTML tags as the tags would bring nothing but bias in our accuracies.

Right after, we renamed the columns for ease of access and then converted all the headlines into smaller-based charactars. Furthermore, we joined the individual features all into one list i.e. making it chunks of paragraphs. A few steps ahead, after converting them into vector features (implementation of Bag of words).

Righteously, we used Random Forest Classifier , constructing a multitude of decision trees at training time, and resulting in the class that is the mode of the average prediction of the individual trees. The implementation pseudo-code is presented below and the main file will be uploaded on Google Colab.



After repeating the same for our training dataset (transformation into paragraphs), we got an accuracy of 52%.

As our initial naive approach yielded quite non-favouring results, we presumed to research online for various other classifiers and analysis frameworks to come up with better features and actually understand the dataset provided.

### **Vectorizing words**

In an attempt to find more accuracy, we chose Tf-Idf vectorizer to create our feature set. Using this we were able to visualize the weight distribution of words. Weight distributions is an effective way to understand the classification. Initially, the weight distribution was leaning more towards the higher end of the spectrum. For example if the range is between 3.6 to 11.7, the mean value was at 10. This showed us a possibility of skewed dataset. Thus, we used features such max\_df, max\_features and n-gram range to further balance the mean.

### **Looking for the best:**

After learning about the data we realized that we would have to use a lot of different feature settings to get the best out of the vectorizer and machine learning model. So in order to get the best model with the best parameters, we intended to use GridSearchCV. This model uses a list of possible different parameters for all the models and then combine them with each other to find the best parameters for better performance. Thus, we used multiple models such as Multinomial Naive bayes, Logistic regression, Random forest, XGBoost, etc. with multiple parameters to get the best parameters.

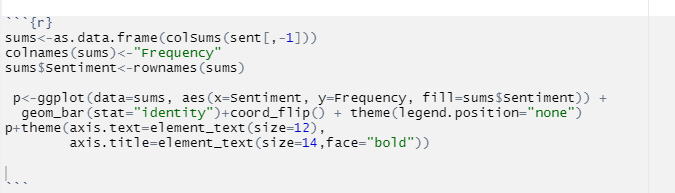
Another approach that we used was using a voting classifier. This classifier uses all the specified models and uses their predictions to vote for the label. For example, if there are 5 models, three models predicted the label to be 1 and other two predicted it to be 0. Then the classifier would classify it as one.

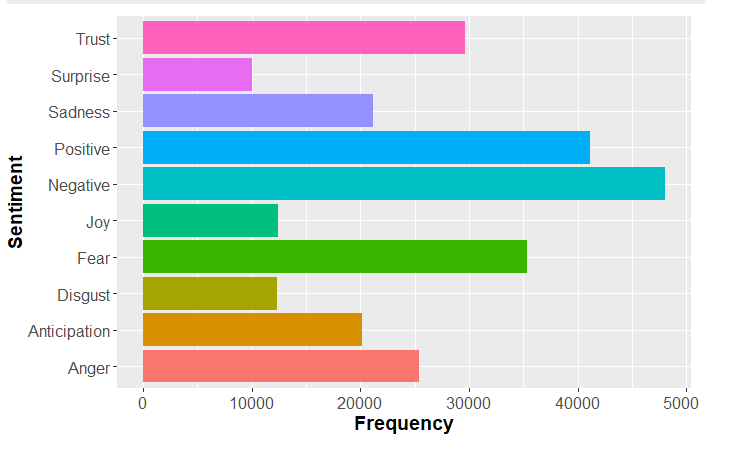
### **Deep learning**

Last approach that we used to further improve our accuracy was through deep learning. We used Google’s BERT to train on the dataset. It uses neural networks and machine learning to teach itself to better understand user queries and tries to understand the context behind words. BERT stands for Bidirectional Encoder Representation of Transformers. “Bidirectional Encoder Representations” means it reads the entire set of words in an input to understand how they interact with each other. “Transformer” is a machine learning model that deals with language processing in artificial intelligence.

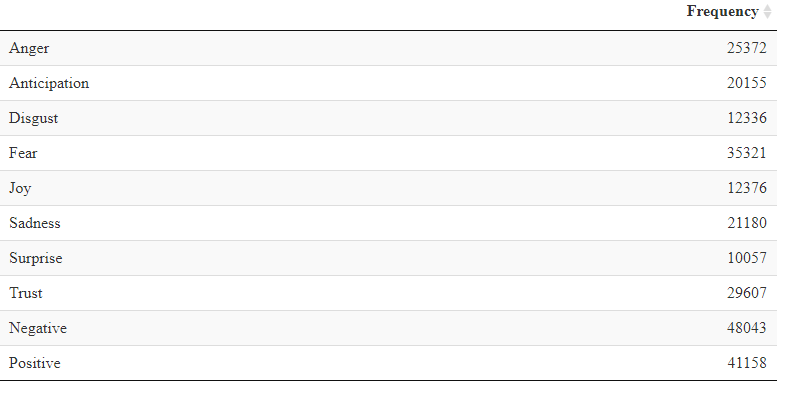
# **Results**

The results are below and code is provided under the name of “475 Project Semantic Analysis.Rmd” and is shown in figure 2 and *figure 3* below.





***Figure 2***



***Figure 3***

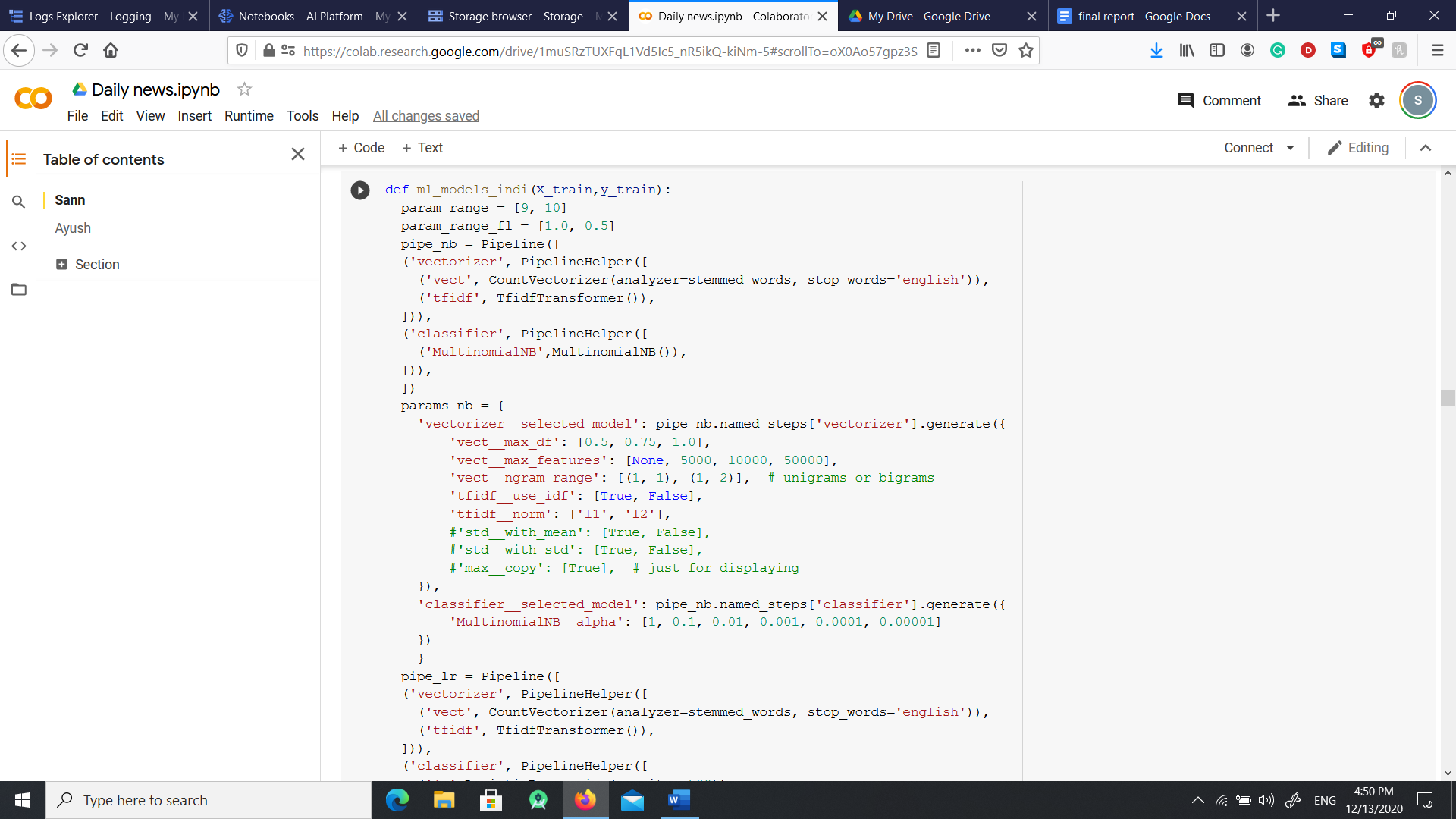
This gave the distribution of sentiments throughout the dataset ultimately implying that the news provided on the reddit news was primarily negative and negative news tends to distribute at the swiffer rate on the internet. Some notable outcomes are mentioned below:

* Russia was mentioned 3537 times, and ISIS is mentioned 31 times.
* Our prediction accuracy was levelled on 52%

As our initial naive approach yielded quite non-favouring results, we presumed to research online for various other classifiers and analysis frameworks to come up with better features and actually understand the dataset provided. It is evident to consider multiple data points in conjunction with each other.

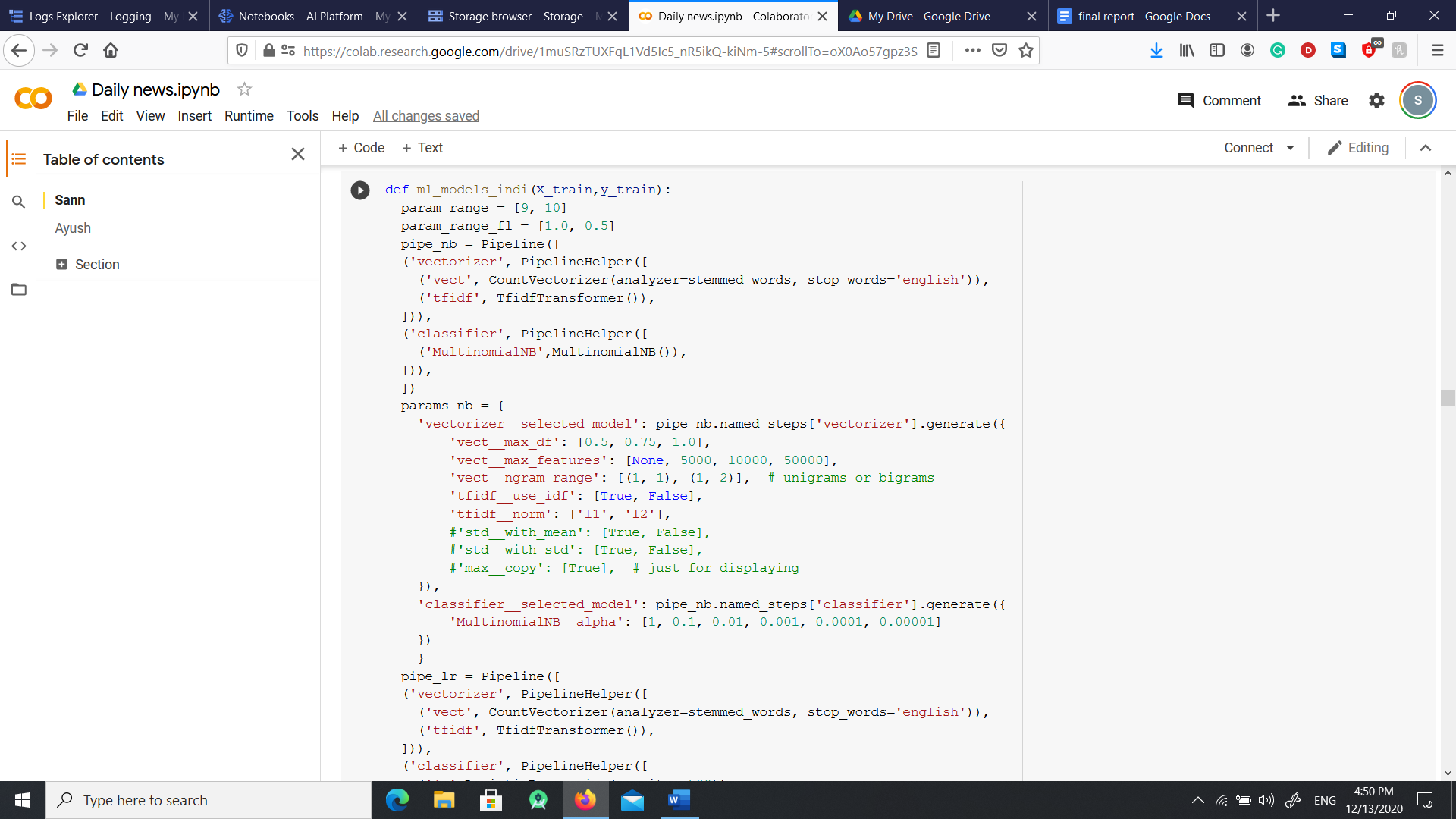
## **Vectorizing Approach**

Our second approach using vectorizer with GridsearchCV to find out the best setting did turn out to be fruitful. To achieve this we created a pipeline to put everything to automate the job. We defined a set of different parameters to combine and find the best parameters.



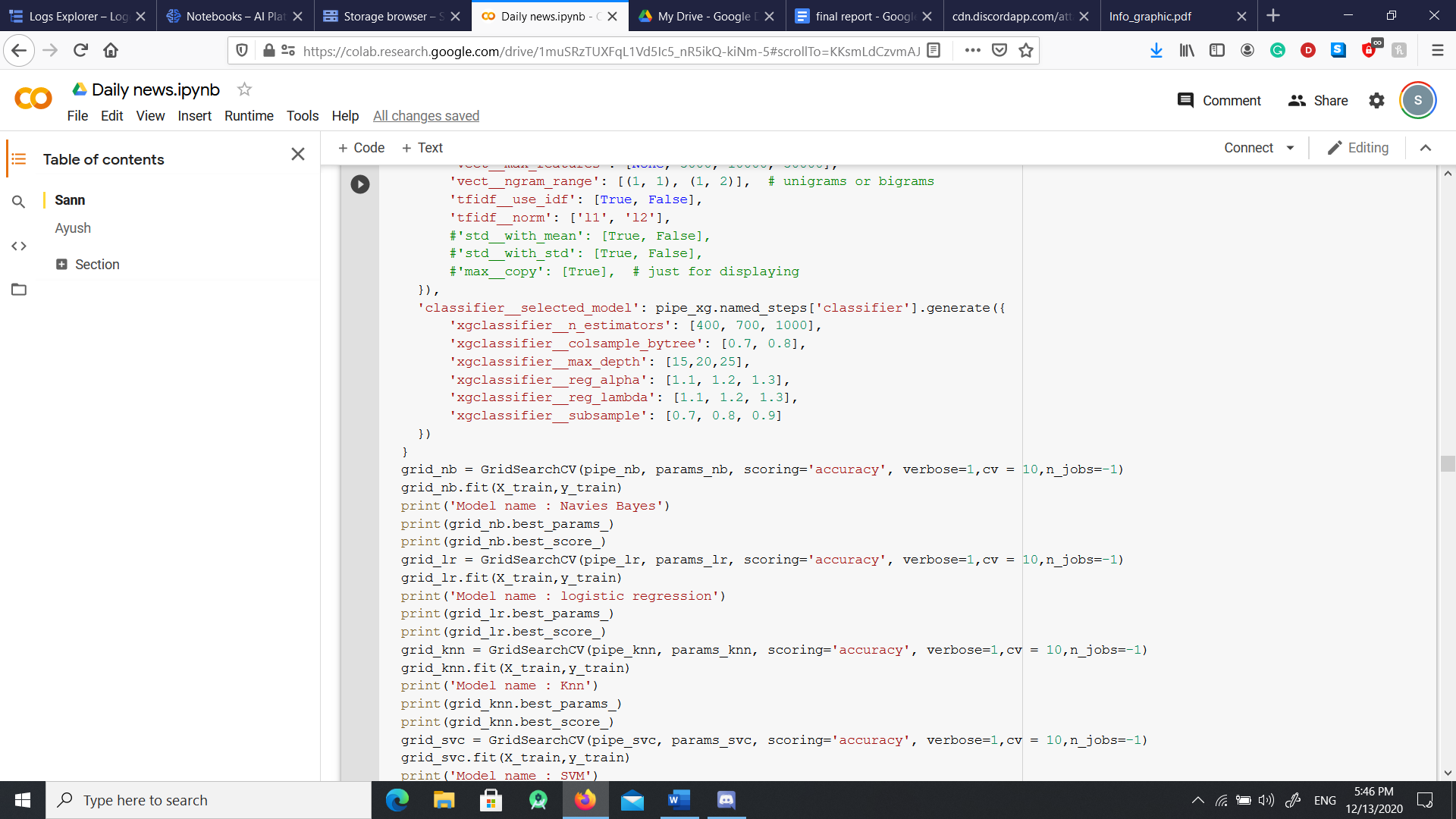
***Snippet of a pipeline***

The above snippet shows the pipeline for the Multinomial Naives Bayes model. In the above code the first line helps in easy classification of vectorizers and classifiers through the use of Pipeline Helper function. In the pipeline helper we define the name and model’s constructor that we want to deploy. For example, Inside Vectorizer pipeline helper we deployed Tf-Idf Transformer by the name of ‘tfidf’.



***Snippet of a defining parameters***

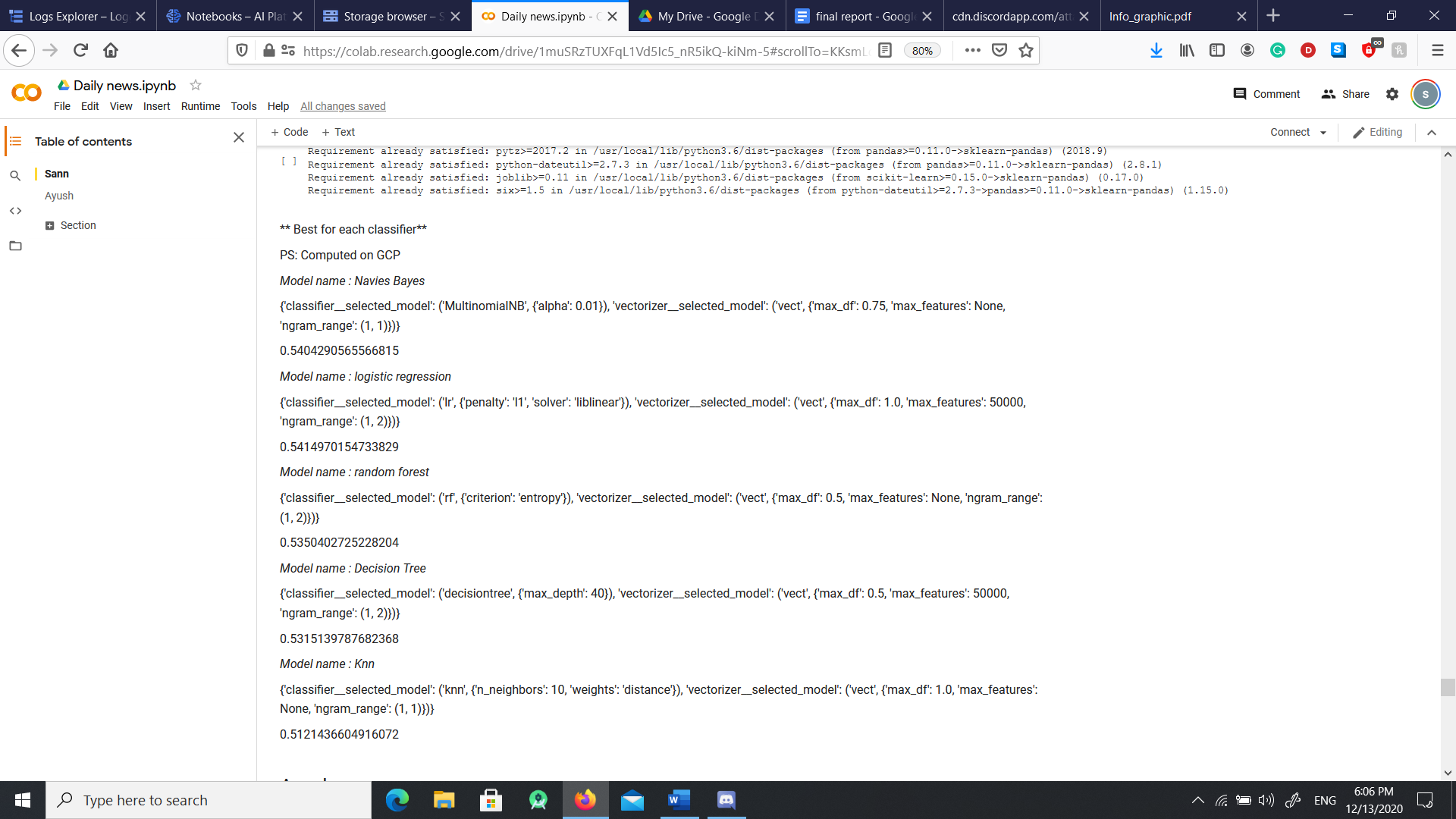
The above snippet shows the different parameters to be combined to evaluate the model on different parameters. These parameters are defined by (model\_name)\_\_(model\_parameter) = [list of different values for the parameters]. For example, 'MultinomialNB\_\_alpha': [1, 0.1, 0.01, 0.001, 0.0001, 0.00001]. Here Multinomial NB model will be evaluated on different values of alpha defined in the list.



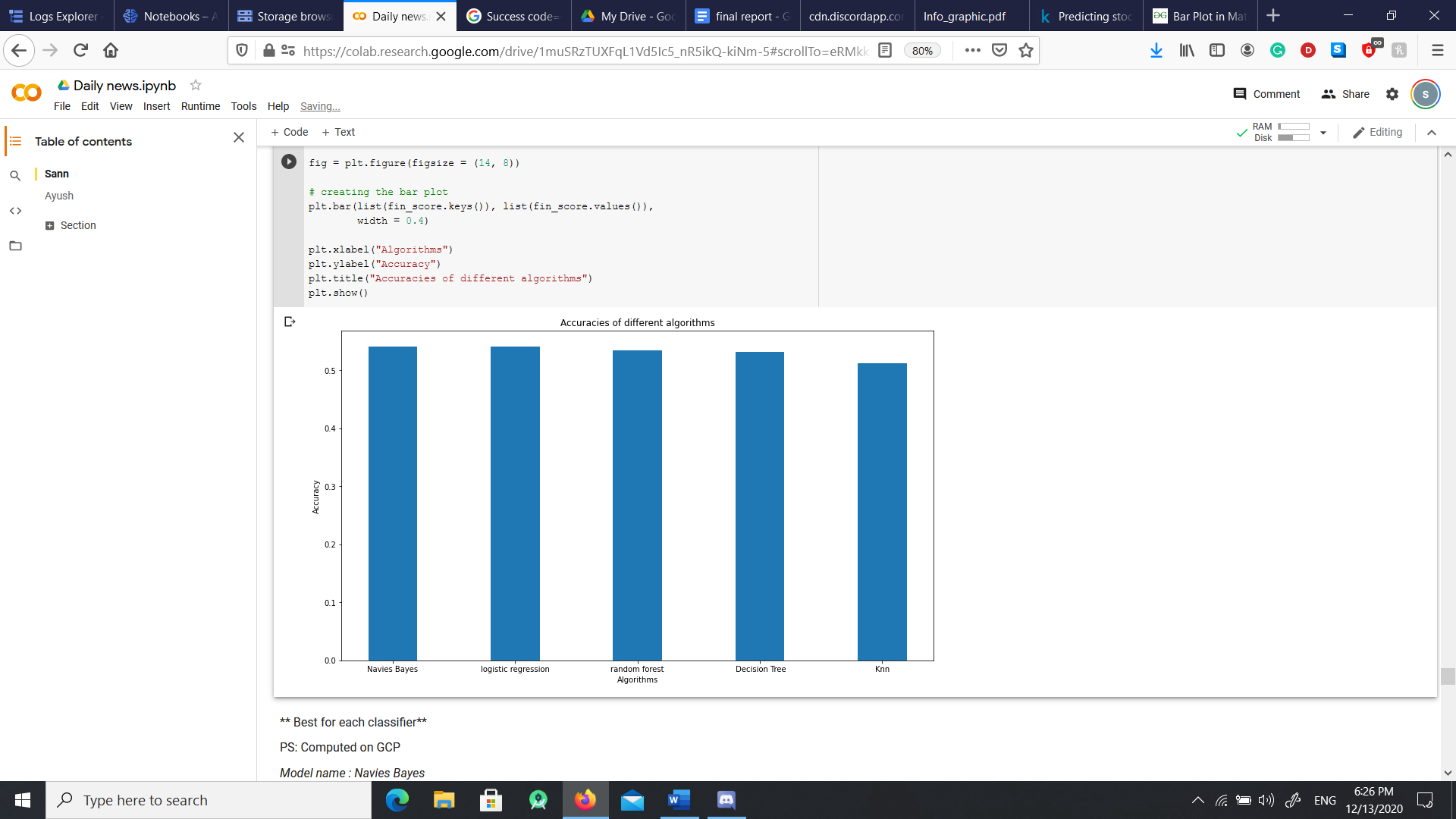
***Snippet of GridSearch function***

Combining the pipeline with parameters happens at the GridsearchCV function. The above snippet shows how to run the function. To initialize, define pipeline, parameters, scoring metric, verbose, # cross validations, number of cores to use for the jobs in the respective order. After initializing fit the models on the dataset. Lastly, get the best performing parameters and its score using best\_params\_ function and best\_score\_ respectively.

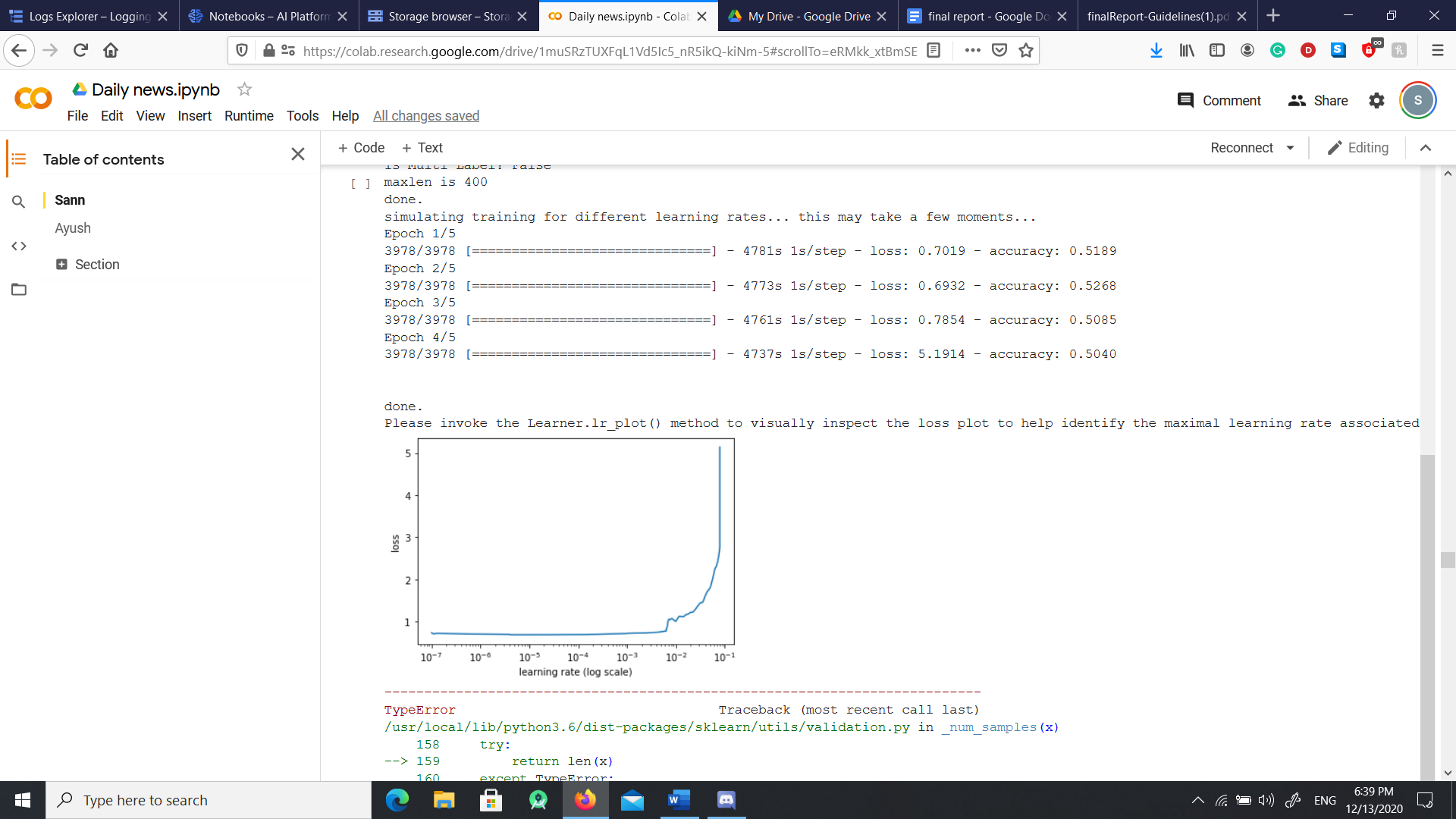
For this job we used a number of classifiers. These classifiers were trained on multiple parameters to get the best out of these classifiers. Originally, we intended to use seven classifiers on various parameters namely, Multinomial Naive Bayes, Knn, logistic regression, support vector machine, decision tree, random forest and XGBoost. Unfortunately, due to resource limitations SVM and XGBoost could not be trained on GridsearchCV. These models could be accounted for our future goal. On the other hand, the remaining five models that we were able to train showed better results than sentiment analysis, further improving our accuracy to 54.14%.



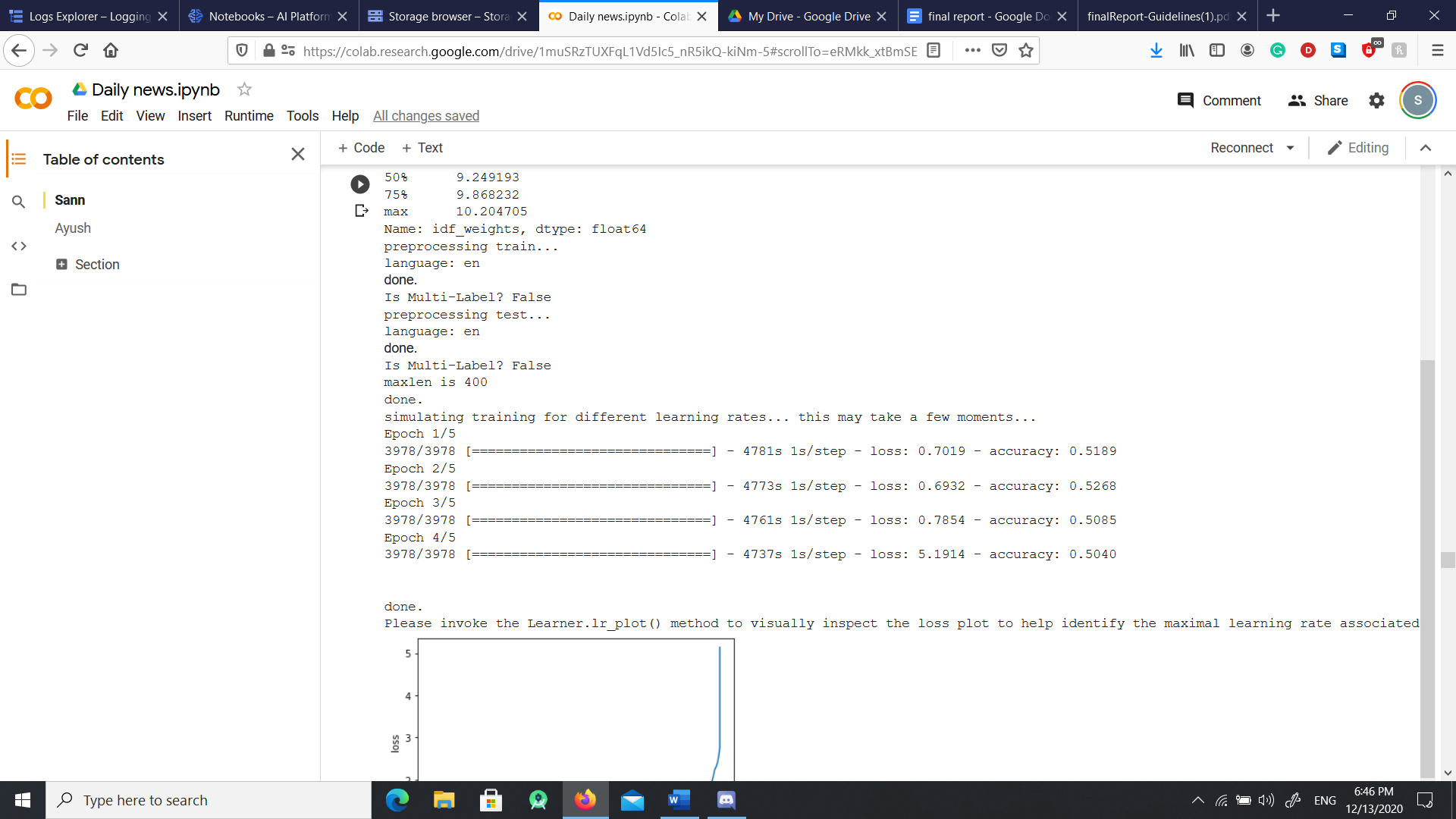
***Result of every classifier with their best parameters and accuracy***



## **Deep Learning**

In order to strengthen our accuracy we turned to google BERT. BERT was fed all the data to further predict the label. Unfortunately in our case bert suffered from a very high loss rate at higher learning rates. Keeping the learning rate very low would take hours for Bert to train. Thus, we simulated bert at different learning rates. 

The result was 52.7% accuracy while having a very high loss of 0.7.



***Results of BERT’s accuracy at different learning rates***

# 

# **Conclusion**

In Spite of the results accumulated, we brought different approaches into force to make this project a worthwhile engagement. Substantial tools and software had been put to use. Some additional ideas to consider for future work on this project, we shed lights upon using strictly the headlines that will be directly affecting the top 30 companies that make up Dow Jones Industrial Average so there is undeviated firsthand dispatch of useful information for prediction purposes. Another approach was to do grid search and test multiple models in a single pipeline, and define a helper class to help interact among the parameters of pipeline elements that arise before the classifier.

We understand that there is still much room for improvement for the prediction algorithm and classification that we would like to optimize in the future and submit to Kaggle. We would also like to test the various different classifications on more news headlines and abstracts that are actually aligned with the fluctuation of the DJIA stock. We would like to incorporate by having more LSTM branches, so the network can be able to make certain decisions solely based on short and long term trends. While researching other stock prediction solutions, we overcame a notebook that indulged in grouping the textual data based on Dates rather than News Headlines which yielded an 87% accuracy. (link is provided below). Although there were some inconsistencies in the approach, it was uplifting to analyze another direction taken within scope.

There is a saying in the stock market, “buy a stock on rumors, and sell them on news” implying that one should invest in stocks primarily when even the market does not know of its development. Even with anticipation and computation of a great amount of data, it is difficult to predict the shift of the market solely on the basis of statements. Earnings cannot predict prices, and socio-political news cannot predict stocks. Such machine learning determinants and accuracies should be accompanied by Human Intelligence in order to govern a superior analysis and outcome.

# **Bibliography**

[1] Link to Google Collab:

https://colab.research.google.com/drive/1muSRzTUXFqL1Vd5Ic5\_nR5ikQ-kiNm-5?usp=sharing

[1] Sun, J. (2016, August). Daily News for Stock Market Prediction, Version 1. Retrieved [Date You Retrieved This Data] from <https://www.kaggle.com/aaron7sun/stocknews>.

[2] Ortner, A. (2020, October 13). Top 10 Binary Classification Algorithms [a Beginner's Guide]. Retrieved December 04, 2020, retrieved from <https://medium.com/@alex.ortner.1982/top-10-binary-classification-algorithms-a-beginners-guide-feeacbd7a3e2>

[3] BorkaBorka 3355 bronze badges, & Vivek KumarVivek Kumar 26.7k66 gold badges6868 silver badges103103 bronze badges. (1967, June 01). An Typeerror with VotingClassifier. Retrieved December 04, 2020, from <https://stackoverflow.com/questions/49761807/an-typeerror-with-votingclassifier>

[4] Goldberger, A. (2019, March 12). Training Neural Networks for binary classification: Identifying types of breast cancer (Keras in R). Retrieved December 04, 2020, from <https://medium.com/duke-ai-society-blog/training-neural-networks-for-binary-classification-identifying-types-of-breast-cancer-keras-in-r-b38fb26a500c>

[5]Ortner, A. (2020, October 13). Top 10 Binary Classification Algorithms [a Beginner's Guide]. Retrieved December 11, 2020, from <https://medium.com/@alex.ortner.1982/top-10-binary-classification-algorithms-a-beginners-guide-feeacbd7a3e2>

[6]Anuj, (2013, April 13). [scikit grid search over multiple classifiers](https://stackoverflow.com/questions/23045318/scikit-grid-search-over-multiple-classifiers).Retrieved December 1, 2020, from <https://stackoverflow.com/questions/23045318/scikit-grid-search-over-multiple-classifiers>