University of Oklahoma

Predicting Electricity Cost Trends Based Upon Twitter Data

CS 5593 Data Mining

Summer 2021

Rhett Green

green.rhett@ou.edu

Predicting Electricity Cost Trends Based Upon Twitter Data

Rhett Green
School of Computer Science
University of Oklahoma
Norman, Oklahoma USA
green.rhett@ou.edu

ABSTRACT

Twitter is one of the most prevalent social media platforms in the world today, with nearly 200 million active daily users voicing their opinions globally. It provides millions of individuals with a voice and a platform to be heard, but the most interesting aspect of twitter for this study, is that the data is public. The possibilities that Twitter provides for data researchers and miners is abundant, endless and often underrated. This plethora of data, opinions and miscellaneous information, is beginning to be utilized by various conductors of new studies in all fields of research. Recently, the potential for data resources that Twitter offers, is used for a multitude of projects like predicting the stock market, mapping road closures caused by natural disasters, predicting elections and much more.

Twitters API has built in tools that allow you to retrieve tweets from a users timeline, search tweets by keywords, and filter real time tweets but not without limitation [1]. The amount of historical tweets you can generate, is limited to a certain time period from present day, so to counter this, a preexisting twitter scraper was used to generate a data set without these limitations. [2]. The resulting dataset could be sorted by time period, without limitation, and geo location which was vital for categorizing tweets by state. The purpose of the study was to see if a direct correlation between the sentiment of tweets in an areas electric prices existed, and whether or not it could be used to predict future trends. To do this a Naive Bayes model was constructed to determine the sentiment of a tweet in real time by classifying each word in the tweet as either negative or positive and generating a number from all the words in the tweet with a positive score being a positive sentiment and a negative score being a negative sentiment. Once this model was created and tested, it was applied to cleaned twitter data sets from different states. If a state had more positive tweets than negative, the electric cost was predicted to be trending down during that time period. If there were more negative tweets than positive tweets the electric cost's were predicted to be rising in that state.

The model was tested on 10 states during 2020-2021 due to time constraints, but it could be scaled to every state or even include other countries in the future and add other years of twitter data since the twitter scraper is not limited by time. The only thing that would limit it, is twitters inception was in 2006, also Twitter didn't always have the enormous traffic it has today. For the first couple years of its existence, data sets might be too small for accurate predictions [3]. The idea behind this study was proof of concept to see if it is a viable model that can make predictions with some degree of reliability. Of the 10 states, the model correctly predicted the trend of electricity costs 7 times giving it a success rate of 70%. For the first attempt at the model, this success rate is promising because the sentiment analysis model can always be improved for better accuracy, so this could prove to be useful in identifying electric cost trends.

KEYWORDS

Electricity Costs, Twitter, Naive Bayes, Twitter Scraper

1 Introduction

This year many states across America were hit with extreme winter weather causing record low temperatures that resulted in higher electricity prices across the country. States like Texas, whose infrastructure wasn't designed for such harsh weathers, were hit the hardest and many lost power and gas. For some, keeping their electricity during the blackout may of turned out to be worse than losing it, due to the state's market driven power grid system. Some homeowners have received bills in the thousands, even as much as 15,000 dollars for a month's worth of power [4]. With the continuing trend of harsher weather every year this problem could grow worse and a way of predicting prices could prove invaluable.

This study gathered tweets from twitter and applied a sentiment analysis on the data with the hypothesis that people's mood regarding electricity had a direct correlation to the trend of prices in their area. The data used to predict these prices was taken from social media app twitter using a scraper to pull tweets related to electricity prices. Once the data sets were generated using keywords they had to be cleaned and preprocessed removing lots of columns of unnecessary data and also cleaning the actual tweets generated to make it better training data for the model. After the preprocessing was complete the next step was generating a Naive Bayes Sentiment Analysis Model that was capable of determining the sentiment of a tweet without a predetermined word bank. There are two forms of sentiment analysis, the first is performed by using large predetermined word banks that have words categorized as either positive or negative and it scans the data to match words and determine an overall sentiment of a tokenized string or set of strings. The problem with this method however is it doesn't adapt to data and is set in stone and also doesn't fulfill the data mining requirements of this project. The second option which was selected is using a machine learning sentiment analysis tool that analyzes text for as either positive or negative by training machine learning tools with examples of emotions in text. These algorithms automatically learn how to detect sentiment without human input and are far more intelligent and adaptive than the first option.

This model was built using a preexisting twitter data set with 10,000 tweets that had already had their sentiment determined by hand [5]. The reason for using this data set was because with the short summer semester there wasn't time to go through enough tweets and identify sentiment by hand to be able to accurately train a Naive Bayes Model. This public data set provided a good training and testing data set that when used for testing returned a 99.99% success rate on the 2000 test tweets. This model was then used to predict the sentiment of tweets on the cleaned scraper data sets. The result was a list of the tweets with their sentiment analysis scores, the further the score was from zero the more confident the model was in its assessment. The sentiments were counted and turned into a majority verdict for each state to predict the states electricity trends. These predictions were then tested against an Electric Choice data set of residential electric cost's for each state from April 2020 to April 2021 [6].

2 Related Work

The idea for this came from an article about using sentiment analysis to predict the stock market [7]. In this article it covered a study that followed tweets in the United Kingdom regarding the ongoing election that created their own sentiment analysis model to identify the tweets as either positive, neutral, or negative and using this information they built a regression model that could use the sentiment of key words to predict the stock market. Unlike previous studies they used hashtags regarding the election and the sentiment of these tweets to predict moves in the stock market with the hypothesis that polling views would cause a direct fluctuation of the market and while they did find a correlation between public mood and the stock market it wasn't enough to provide a statistical advantage for prediction. The difference between this study and the UK study that should make it more effective is the fact that electric prices can be split into two categories, up or down, so in theory it's easier to predict than the stock market and it was also easier to test the results of the model.

Another issue considered was while using a scraper normally it can only retrieve as far back as three weeks due to twitter API. After some research an article that has a method of retrieving data as far back as needed using Scrapy, an open source and collaborative framework for extracting data from websites written in Python, which enhances the power of scraping engines to obtain an unlimited volume of tweets bypassing date ranges limitation [8]. Their study was purely on the twitter scraping aspect but it provided more knowledge for selecting the scraper twint which was used for this study. This made it easier to generate large twitter data sets flexibly with quick efficiency.

A study from Annamalai University constructed by Dr. M.Govindarajan used a hybrid method of Naive Bayes and a Genetic algorithm to predict sentiment of movie reviews [9]. His findings returned 91.15% accuracy for Naive Bayes, 91.25% for Genetic Algorithm, and 93.80% for his proposed Hybrid NB-Ga Method. After reading the paper and the complexities of the algorithm's the time available for the study meant a hybrid algorithm was out of the question and the Naive Bayes was both accurate and simpler to implement than the Genetic Algorithm so it was selected to be the one implemented for the sentiment analysis on the generated twitter data.

Naive Bayes is a relatively simple yet effective machine learning algorithm that sticks to its name in its naive behavior. Meaning to maintain its efficiency and relative simplicity it has to make lots of assumptions. A study constructed by MIT sought to improve the naive nature of the algorithm by correcting skewed data bias, parameter estimation and weight magnitude errors. Their implementation performed better at handling the more intricate nature of written text because a computer often struggles to handle subtle human innuendos that are hard to code into an algorithms learning pattern and although coding this sort of algorithm was beyond the scheme of this study it provides further efficiency that can be added in the future to improve the prediction rate of the study.

3 Scraper

The first step involved was generating a data set for the model to use for predictions. To do this a web scraper was constructed which is the process of importing information from a website into a local file saved on your computer and it presents one of the most efficient and flexible ways to get data from the web in real time or from historical records. Scraping is not only efficient but can help you gather different data into one place allowing you to take unstructured, scattered data from multiple sources and collect it in one place and make it structured for further use. As discussed before they are prebuilt into Twitters API, application programming interfaces, which is essentially the way computer programs interact with each other so that they can request and deliver information. What makes Twitter so unique is that its data is public and their API platform provides broad access to the data that users have chosen to share and allows people to build software that directly integrates with Twitter. Therefore they technically have a scraper built in that can be used if you get authentication from Twitter that allows you to pull data directly from their servers however it has a a rate limit. Your rate limit depends on your type of authentication like for instance user authentication is any OAuth process that is based on the user. Application-only authentication is for your application credentials and not based on user so for searching, you have a 15 minute window of 180 tweets for user and 450 tweets for app. Not only would this be highly limiting but gaining authorization takes time that was not available for the time table of this study. So too circumvent this issue and the API all together twint was used for the twitter scraper.

Twint is a preexisting Twitter scraping & OSINT tool written in Python that doesn't use Twitter's API, allowing you to scrape twitter data while evading most API limitations [2]. Its simple to implement and most importantly involves no initial Sign-in or Sign up and has no limit of downloading tweets. Installing twint is as simple as running the command pip install twint and importing into the python file. There were a few features of twint that made it ideal for grabbing data for this study, its geo feature and date range feature. These allow you to specify coordinates and a radius of a circle around those coordinates to pull tweets from because some tweets include the location the tweet was sent from which was useful in classifying data by state. The next feature were the commands ".Until" and ".Since" this allowed the scraper to pull tweets only in a specified time frame in this case 2020 to 2021. To make sure the tweets are about a certain subject the ".Search" feature is used to give the scraper keywords to search for in tweets, and by using the word "OR" between different words it allows the scraper to search for a list of words in this case the search keywords are: ' "electric prices" OR "electricty cost" OR "electricty bill" OR "enery cost" OR "energy grid" OR "electric bill" OR "utility price" OR "utility cost" OR "utility bill" OR "electric cost" OR "electric grid" OR "power outage" OR "electricity" '. Using these inputs the scraper generated data from twitter that matched the requirements and the resulting tweets were returned to a csv file with 36 columns: id, conversation_id, created_at, date, time, timezone, user_id, username, name, place, tweet, language, mentions, urls, photos, replies_count, retweets_count, likes_count, hashtags, cashtags, link, retweet, quote_url, video, thumbnail, near, geo, source, user_rt_id, user_rt, retweet_id, reply_to, retweet_date, translate, trans_src, trans_dest. Each row contains a tweet with its information filling these 36 columns, for the 2020-2021 time period Texas generated 12,131 tweets, Hawaii generated 271 tweets, Idaho generated 51 tweets, Kansa generated 1,233 tweets, Louisiana generated 5,286 tweets, Arizona generated 631 tweets, Utah generated 74 tweets, Wisconsin generated 481 tweets, California generated 7,883, and Massachusetts generated 1500 tweets. Now that the data had been collected the next step was preparing the data for the Naive Bayes model.

4 Preprocessing

The data file created from the scraper was a very large, information-dense csv file with lots of information not needed for sentiment analysis. Pre-processing is vital because raw tweets without pre-processing are highly unstructured and contain redundant and often problematic information. Removing unnecessary noise, characters, and syntax improves the accuracy of the model during training. The original csv file contained 36 columns. This was reduced to one after removing obsolete information, such as rows with "N/A" values to prevent skewing the data. The only column needed for sentiment analysis was the tweet column that contained the actual text submission of the tweet. Next, any duplicates of the data were dropped to ensure the data didn't become skewed by repeating tweets that could be generated by bots. Then, began the process of cleaning the actual tweets.

The first step was reading through tweets and looking for non textual information that could throw off the model, as it is only designed for reading words or emoticons. After reading through the tweets, twitter handles in the form of @mention, hashtags, weblink, audio and video tags, and many types of punctuation symbols were all observed. So cleaning the data involved removing any web links, whether that be "http" or "bitly" links, and also removing any pictures. Next, nonessential information such as retweet, @user, and hashtags were removed as it was not important to the model. Then, after any audio or video tags were removed, the data was ready for lemmatization and tokenization. Lemmatization is the process of changing words to their base form. This could have been accomplished through stemming or lemmatization. Lemmatization considers the context and converts the word to its meaningful base form. However, stemming just removes the last few characters, often leading to incorrect meanings and spelling errors. After this was complete and every word was as simple as it could be, the remaining strings were tokenized. This is essentially the process of dividing text into a set of meaningful pieces, or "tokens". In this case, sentences were turned into sets of strings where each word is a token that the model can use for training to determine if it is a positive or negative word.

A database of stop words was also used to remove filler words like "a", "the", and, "etc" to make the data set more meaningful and cost effective in relation to computing time as these words are useless for natural language processing and could dilute the sentiment scores [10]. Once all this was complete it was called in a method tokenizeTweets to apply all these changes and return a data frame. For example, "Thanks @GoGriddy for cutting my electric bill in half. We were paying $95 / month with @reliantenergy and now with @GoGriddy we are paying ~$45 / month. https://t.co/Ws3IXbczoq" was returned as "thank cut electric half pay month pay month".

For the thought process behind this process and what information to exclude I relied heavily on the work done in an article about the role of text pre-processing in twitter sentiment analysis by the National Institute of Technology in Hamirpur, India [11]. Now that there was clean

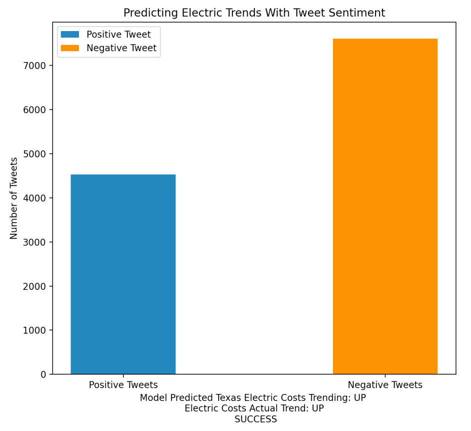

Figure 1: Barplot displaying the results of Texas sentiment results and prediction which was correct.

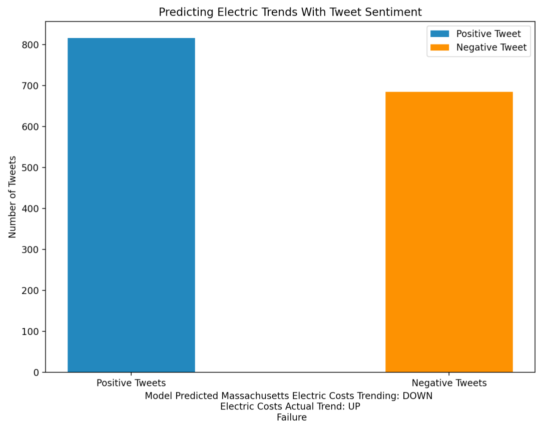

Figure 2: Barplot displaying the results of Massachusetts sentiment results and prediction which was incorrect.

data for a model to run on, it was time to build the Naive Bayes model.

5 Naive Bayes Model

Naive Bayes is a classification technique based on Bayes' Theorem that assumes independence among predictors. In layman terms, the Naive Bayes classifier makes the assumption that the presence of a particular feature in a class is unrelated to the presence of any other feature. This is useful for this study because it is easy and fast to predict the class of the test data sets and it also performs well in multi class prediction while remaining easy and fast at making predictions. It also has great noise resilience. Naive Bayes thrives in differentiating between irrelevant features and its prediction capabilities won't be seriously affected like some other algorithms. After reading a paper by the University of Montreal on the different types and aspects of Naive Bayes, it became clear that one of the issues with the algorithm is its inability to understand sarcasm and non literal literature that humans often use but a computer has trouble recognizing [12]. If someone were to tweet "What a great presentation…. not!" after tokenization the tweet would become "great presentation not" and Naive Bayes would see the word "great" which is usually found in positive tweets so it would return positive sentiment, although the tweet had a negative connotation. This lack of intelligence hurts simpler algorithms. However, the paper did continue on to show the overall accuracy of the algorithm is still high and is the most efficient algorithm with such simplicity that could handle large data sets without bogging down the ram so for the purpose of this study it was perfect.

Ideally, a large testing and training set assessed by hand would improve construct validity. However, this was unrealistic due to time constraints. To overcome this, an outside data set had to be used and luckily the Natural Language ToolKit or NLTK had a prebuilt twitter data set for generating models [10]. Its tweets were classified into two different files "positive_tweets.json" and "negative_tweets.json. This predetermined sorting made it ideal for modeling the Naive Bayes Classifier. Each data set was cleaned and 4,000 tweets from both the positive and negative tweets were assigned to the training set and the remaining 1000 of each were assigned to the test set. Therefore, the total training data set contained 8000 tweets and the test set contained 2000.

Next, a frequency table counted how many times a word appeared in either positive or negative tweets. To do this, a binary array was created for both data sets' training and testing, that assigned a 1 to tweets that were in the positive list and a 0 to tweets that were in the negative list. The frequency table method went through the data sets word by word and counted the amount of times a word occurred in positive tweets and negative tweets, adding the count for the word to the frequency dictionary. Next the actual training of the model began using the frequency table. The first step was calculating the number of different words in an entry. Then, each word's sentiment probability was calculated using the frequency of that word in negative tweets and positive tweets. For each one positive and negative, a probability

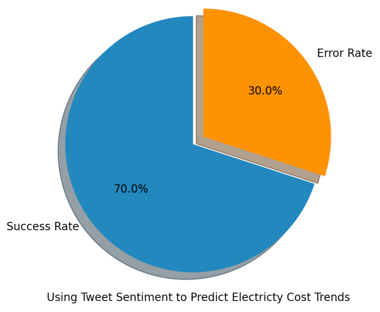

Figure 3: Pie Chart displaying the accuracy of the prediction of the 10 states tested.

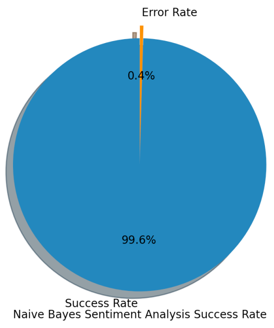

Figure 4: Pie Chart displaying the accuracy of the sentiment analysis on the test data set.

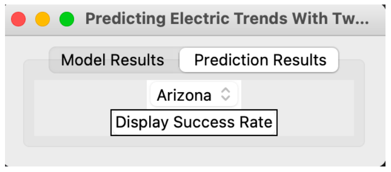

Figure 6: Prediction Results tab selected that displays data regarding electric cost trends predictions.

was calculated by adding the frequency by one and dividing by the number of positive or negative words available added to the length. The resulting equation looked like: probPos = (frequencyPositive + 1) / (numPos + length). Finally the probability of a words sentiment was returned by performing a logarithm of the division of the probPos by the probNeg.

After the training of the model, sentiment predictions were made to test the accuracy. This was done by reading a tweet and accumulating the individual word sentiment probability generated from the training. If the resulting number was negative, the tweet was considered negative. If the resulting number was positive, it was considered a positive tweet. The greater the distance from zero (negative or positive), the more confident the model was in its prediction. The results returned a tweet and its score like such: “MY PUPPY BROKE HER FOOT :(-> -10.35”. This tweet was returned as strongly negative. To check the accuracy of this model it was run on the negative test set and all the negative and positive predictions were counted, the same was done for the positive set. Out of the 2000 test tweets, the model correctly identified 1,991 with only 9 errors. This is a 99.996% success rate providing statistical evidence that this model was highly accurate.

6 Results

The method used to check the results of this study was accuracy, and this was measured by the effectiveness of the electric cost trends predictions. Using the scraper to generate the data, then preprocessing it to clean it, along with the NLTK data, then training the sentiment analysis model, all led to applying the model on the scraper generated data. Once the data had been read in to a CSV and converted to a data frame, preprocessing removed the unnecessary columns and information to prepare the data for the next step; sentiment analysis. The model operates on the twitter generated data in the same way it was applied to the test data returning the predicted sentiment of each tweet but this time a counter was included to count the number of positive and negative tweets recorded for each state data set.

Now that the counter had the statistics it was time for predictions, the idea behind predictions was simple if a state returns more positive tweets than negative the states view on electricity during that time period is positive, so the electricity prices are trending down. If there are more negative tweets than positive it is predicted that the electric prices are rising causing sentiment of electricity in that area to be negative. To check the results the data set generated by electric choice that showed each states electric cost trends from 2020-2021 were cross checked with the results of the models prediction [6]. To display the results a GUI was constructed with two tabs “Model Results” and “Prediction Results” shown in Figures 5 and 6. Model Results has two buttons “Display Data” and “Display Success Rate” the first of which displays a histogram with the number of correctly identified tweets from the test set the second button displays the success rate using a pie chart. The second tab has a drop down menu and a “Display Success Rate” button. The first of which contains the ten states the model was tested on, by selecting one, a histogram is shown displaying the states number of positive tweets and negative tweets along with its prediction and if the prediction was correct or not. Figure 1 and Figure 2 are both good examples of this with Figure 1 being Texas and returning a successful prediction, and Figure 2 being Massachusetts, which returned a Failure result for its prediction. The Display Success Rate button displays a piechart of the success rate of predictions. For the 10 states the model was tested on, it successfully predicted 7 scoring a 70% success rate. The models effectiveness was very promising for the first attempt and provides a proof of concept that tweets can be used to predict electric costs in an area ,but there is always room for improvement.

One of the future improvements this study could benefit from would be improving the sentiment analysis model to better identify the more intricate subtleties of the human language. After reading a paper on the effectiveness of current sentiment analysis models on sarcasm and irony, it’s apparent that although improvements have been made, the area still needs more research until a model is effective. When that time comes, it could be added to this study to improve the accuracy of the sentiment analysis [9]. Another helpful paper that could have relevance for future work, used tweet sentiment to make predictions of bitcoin prices, but they also added volume of tweets to their model for increased accuracy [13]. This method could be added to this study to help improve accuracy, but would need further testing to see how much improvement it could provide. The final thing that could greatly improve the results of this study, would be increasing the study size to all states. The reason this study included only these 10 states, was the short nature of the summer semester it took place during. The states chosen were the ones with the greatest increase or decrease, so it would be interesting to see the prediction success rate of states with less change. Would this stability lead to less tweets and therefore data to perform sentiment analysis on? The results of the study despite the short semester are promising, the goal was to prove if there is a correlation between tweet sentiment and electric price trends. This study proved that building a viable model is possible. For future models built on this proof of concept, adding some of the additions mentioned above could improve the model.

7 Conclusion

This research introduces a new method of predicting the trend of stock prices using the sentiment of tweets gather via data scraper. This study combined multiple methods from previous work using sentiment classifiers and scrapers to create a unique model.

An experiment was built to test the models predictions on 10 states these were the states with the highest change during 2020-2021. The results for these chosen states was promising with an accuracy rate of 70%. In conclusion this model provides a promising proof of concept that with additional testing and work could prove an accurate model for predicting electric cost trends.

References

[1] Anon. About Twitter's APIs. Retrieved August 4, 2021 from https://help.twitter.com/en/rules-and-policies/twitter-api

[2] Twintproject. Twintproject/Twint: An advanced Twitter scraping & OSINT tool written in Python that doesn't use Twitter's API, allowing you to scrape a user's followers, following, tweets and more while evading most API limitations. Retrieved August 4, 2021 from https://github.com/twintproject/twint

[3] Anon. Twitter usage statistics. Retrieved August 4, 2021 from https://www.internetlivestats.com/twitter-statistics/

[4] Anon. U.S. Energy Information Administration - EIA - Independent Statistics and Analysis. Retrieved August 4, 2021 from https://www.eia.gov/todayinenergy/detail.php?id=47876

[5] Anon. For Academics - Sentiment140 - A Twitter Sentiment Analysis Tool. Retrieved August 4, 2021 from http://help.sentiment140.com/for-students/

[6] Anon. Instantly see the electric rates and plans available to your home or business: Retrieved August 4, 2021 from https://www.electricchoice.com/electricity-prices-by-state

[7] Author links open overlay panelTahir M.NisarManYeung, Tahir M.Nisar, ManYeung, and AbstractIn order to explore the relationship between politics-related sentiment and FTSE 100 movements. 2018. Twitter as a tool for forecasting stock market movements: A short-window event study. (February 2018). Retrieved August 4, 2021 from https://reader.elsevier.com/reader/sd/pii/S2405918817300247?token=45EFE1D315E3C7D5F9478D674AB40F3526310F17C704DFA50A37BB8ED8D0D409A9C3621BAE9A6E8BC13DB3380885C352&originRegion=us-east-1&originCreation=20210623010616

[8] A. Hernandez-Suarez, G. Sanchez-Perez, K. Toscano-Medina, V. Martinez-Hernandez, V. Sanchez, and H. Perez-Meana. 2018. A web scraping methodology for bypassing twitter api restrictions. (March 2018). Retrieved August 4, 2021 from https://arxiv.org/abs/1803.09875

[9] Rennie, Shih, Teevan, Teevan (2004). Tackling the Poor Assumptions of Naive Bayes Text Classifiers, Massachusetts Institute of Technology, Cambridge, MA 1–8.

[10] Anon. 2021. Removing stop words with NLTK in Python. (May 2021). Retrieved August 4, 2021 from https://www.geeksforgeeks.org/removing-stop-words-nltk-python/

[11] Singh, Kumari, (2016). Role of Text Pre-Processing in Twitter Sentiment Analysis, National Institute of Technology, Hamirpur 177 005, India, 1–6

[12] Rish (2001). An Empirical Study of the Naïve Bayes Classifier, Université de Montréal, 1-6

[13] Paul Simpson. 2018. LSTM model predicting Bitcoin with tweet volume & sentiment. (November 2018). Retrieved August 4, 2021 from https://medium.com/@DrPaulSimpson/lstm-model-predicting-bitcoin-with-tweet-volume-sentiment-bc3c490271a7