# Impact of Cyber-attack on stock price of a publicly traded company

**Team:**

**Gaurav Jain  
Kartik Kapadia  
Kaushal Shah  
Pradeep Yadav  
Vaibhav Shah**

1. **Requirement Gathering**

**Objective**  
One of the goals of this project is to analyze news articles related to a few publicly traded companies that have been the target of cyber-security attacks. In quite a few cases the cyber-attacks may have occurred for weeks or months before it was detected and counter-measures were put in place. It will be interesting to compare counts of news articles about a company related to cyber security or attacks prior to the company releasing information about a cyber-attack, and the counts of such articles after the company releases this information.

For this project we plan to analyze news articles for approximately four months; two months before the company released information about the attack and two months after. Each news article will be classified as being related to cyber security/attack or not by analyzing the content of the article. Counts of cyber security related articles for each day in the four-month period will be tracked.

For the same four-month period, the stock price of the company will be analyzed. It will be interesting to see if there is any correlation in the change of stock price to the number of the cyber-security related articles.

1. Change, if any, to the number of articles related to cyber security for the company that was attacked over the four-month period being considered.

 2. Change, if any, to the stock price of the company during the same time period.

1. Conclude if there is a significant impact of the cyber-attack to the stock price of the company.

Following is the list of companies we are planning to perform sentiment analysis on and keep track of the impact of cyber breach on their stock prices.

**Anthem- February 04, 2015(disclosed)**

**Sony Pictures- November 24, 2014**

**Apple iCloud hack – October 20th 2014**

**Home Depot – November 6, 2014**

**JP Morgan Chase- September 15, 2014**

**Target- December 18, 2013**

**Paypal – December 3, 2010**

**Adobe systems- October 3, 2013**

**Citigroup- June 27, 2011**

**Experian- October 1, 2015**

1. **Data Collection**

**New York Times Article Search API**

**Identify source of data**With the Article Search API v2, you can search New York Times articles from Sept. 18, 1851 to today, retrieving headlines, abstracts, lead paragraphs, links to associated multimedia and other article metadata.

Note: In URI examples and field names, *italics*indicate placeholders for variables or values. Brackets [ ] indicate optional items. Parentheses ( ) are not a convention — when URIs include parentheses, interpret them literally.

|  |  |
| --- | --- |
| **The Article Search API at a Glance** | |
| **Base URI** | http://api.nytimes.com/svc/search/v2/articlesearch |
| **HTTP method** | GET |
| **Response formats** | JSON (.json), JSONP (.jsonp) |

**URI STRUCTURE**Article Search requests use the following URI structure:

http://api.nytimes.com/svc/search/v2/articlesearch.response-format?[q=search term&fq=filter-field:(filter-term)&additional-params=values]&api-key=####

Example Request URI to search articles containing term “walmart” within the period 01/01/2015 to 10/31/2015:

<http://api.nytimes.com/svc/search/v2/articlesearch.json?q=walmart&begin_date=20150101&end_date=20151031&api-key=sample-key>

Response for the above query in JSONformat (with key-value pairs):

{   
            "web\_url":"http:\/\/www.nytimes.com\/2015\/10\/15\/business\/walmart-sales-forecast-share-buyback.html",   
            "snippet": "The retailer also said its heavy investment in wages and e-commerce would weigh on future earnings.",   
            "lead\_paragraph": "The retailer also said its heavy investment in wages and e-commerce would weigh on future earnings.",   
            "abstract": "Walmart upsets investors by saying sales will be flat in 2015; shares fall 10.04 percent to $60.03, largest drop in 17 years; cites big investments in wages and e-commerce competition as factors.",   
            "print\_page": "1",   
            "blog": [],   
            "source": "The New York Times",   
            "multimedia": [{   
                "width": 190,   
                "url": "images\/2015\/10\/15\/business\/15walmart-web1\/15walmart-web1-thumbWide.jpg",   
                "height": 126,   
                "subtype": "wide",   
                "legacy": {   
                    "wide": "images\/2015\/10\/15\/business\/15walmart-web1\/15walmart-web1-thumbWide.jpg",   
                    "wideheight": "126",   
                    "widewidth": "190"   
                },   
                "type": "image"   
            }, {   
                "width": 600,   
                "url": "images\/2015\/10\/15\/business\/15walmart-web1\/15walmart-web1-articleLarge.jpg",   
                "height": 400,   
                "subtype": "xlarge",   
                "legacy": {   
                    "xlargewidth": "600",   
                    "xlarge": "images\/2015\/10\/15\/business\/15walmart-web1\/15walmart-web1-articleLarge.jpg",   
                    "xlargeheight": "400"   
                },   
                "type": "image"   
            }, {   
                "width": 75,   
                "url": "images\/2015\/10\/15\/business\/15walmart-web1\/15walmart-web1-thumbStandard.jpg",   
                "height": 75,   
                "subtype": "thumbnail",   
                "legacy": {   
                    "thumbnailheight": "75",   
                    "thumbnail":"images\/2015\/10\/15\/business\/15walmart-web1\/15walmart-web1-thumbStandard.jpg",   
                    "thumbnailwidth": "75"   
                },   
                "type": "image"   
            }],   
            "headline": {   
                "main": "Walmart Stock Sinks After a Warning on Sales",   
                "print\_headline": "Walmart Cuts Back Its Forecast "   
            },   
            "keywords": [{   
                "rank": "1",   
                "is\_major": "Y",   
                "name": "organizations",   
                "value": "Walmart Stores Inc"   
            }, {   
                "rank": "2",   
                "is\_major": "N",   
                "name": "subject",   
                "value": "Shopping and Retail"   
            }, {   
                "rank": "3",   
                "is\_major": "N",   
                "name": "subject",   
                "value": "Company Reports"   
            }, {   
                "rank": "4",   
                "is\_major": "N",   
                "name": "subject",   
                "value": "E-Commerce"   
            }, {   
                "rank": "5",   
                "is\_major": "N",   
                "name": "subject",   
                "value": "Stocks and Bonds"   
            }],   
            "pub\_date": "2015-10-15T00:00:00Z",   
            "document\_type": "article",   
            "news\_desk": "Business",   
            "section\_name": "Business Day",   
            "subsection\_name": null,   
            "byline": {   
                "person": [{   
                    "organization": "",   
                    "role": "reported",   
                    "firstname": "Hiroko",   
                    "rank": 1,   
                    "lastname": "TABUCHI"   
                }],   
                "original": "By HIROKO TABUCHI"   
            },   
            "type\_of\_material": "News",   
            "\_id": "561e894b798810299d3874b7",   
            "word\_count": "1125",   
            "slideshow\_credits": null   
        }

**Accessing the Data**Following is the code snippet (generating above response) which connects to this API and construct Article objects which are then written to a file in the specific published date folder:

# private static NYTAPIKey *apiKey* = newNYTAPIKey("c807d4d1c6f9af62ede222b642648339:0:73319947");

# private static String *ARTICLE\_SEARCH\_BASE\_URL* ="http://api.nytimes.com/svc/search/v2/articlesearch.json";

# private JSONObject responseJSON;

# public ArticleSearchQueryImpl(NYTAPIKey key) {

# *apiKey* = key;

# }

**The Main Method**

**public** **static** **void** main(String[] args) {

ArticleSearchQueryImpl a = **new** ArticleSearchQueryImpl(*apiKey*);

Query query = **new** QueryImpl();

query.setQuery("walmart");

query.setBegindate("20150101");

query.setEnddate("20151031");

query.setPage(1);

ArticleSearchResponse response = a.query(query);

List<Article> articlesList = response.getArticles();

**for**(**int** i=0;i<articlesList.size();i++){

Article article = articlesList.get(i);

*writeArticleToFile*(article,query.getQuery());

}

**int** noOfPages = response.getTotal()/10;

query=**null**;

response=**null**;

articlesList=**null**;

**for**(**int** i=2;i<=noOfPages;i++){

query = **new** QueryImpl();

query.setQuery("walmart");

query.setBegindate("20150101");

query.setEnddate("20151031");

query.setPage(i);

response = a.query(query);

articlesList = response.getArticles();

**for**(**int** j=0;j<articlesList.size();j++){

Article article = articlesList.get(j);

*writeArticleToFile*(article,query.getQuery());

}

}

}

@Override

**public** ArticleSearchResponse query(Query query) {

**try** {

StringBuilder queryVal = **new** StringBuilder();

queryVal.append(pBase());

**if**(query.getQuery()!=**null** && !("".equals(query.getQuery()))){

queryVal.append(pQuery(query.getQuery()));

}

**if**(query.getBegindate()!=**null** && !("".equals(query.getBegindate()))){

String dateVal = "&begin\_date="+query.getBegindate();

queryVal.append(dateVal);

}

**if**(query.getEnddate()!=**null** && !("".equals(query.getEnddate()))){

String dateVal = "&end\_date="+query.getEnddate();

queryVal.append(dateVal);

}

**if**(query.getPage()>0){

String pageVal = "&page="+query.getPage();

queryVal.append(pageVal);

}

queryVal.append(pAPIKey());

**return** query(**new** URL(queryVal.toString()));

} **catch** (Exception e) {

e.printStackTrace();

**return** **null**;

}

}

**private** ArticleSearchResponse query(URL url) {

System.out.println(url.toString());

ArticleSearchResponse searchResult = **null**;

**try** {

responseJSON = **new** JSONObject(HttpRequest.request(url));

System.out.println(responseJSON.toString());

**if** (**null** != responseJSON) {

searchResult = **new** JSONFactoryImpl().searchResponseFromJSON(responseJSON);

}

} **catch** (Exception e) {

e.printStackTrace();

}

**return** searchResult;

}

**private** String pBase() {

**return** ARTICLE\_SEARCH\_BASE\_URL;

}

**private** String pAPIKey() {

**return** "&api-key=" + apiKey.getKey();

}

**private** String pQuery(String query) {

**return** "?q=" + query;

}

**public** **static** **void** writeArticleToFile(Article article,String companyName){

**try**{

String articleDate=article.getDate();

String filePath="C:\\BigDataProject\\Data\_"+companyName+"\\"+articleDate;

File file = **new** File(filePath);

**if** (!file.exists()) {

file.mkdirs();

filePath=filePath+"\\"+"article1.txt";

}

**else**{

**int** noOfFiles = file.listFiles().length+1;

filePath=filePath+"\\"+"article"+noOfFiles+".txt";

}

System.*out*.println(filePath);

file=**new** File(filePath);

FileWriter fw=**new** FileWriter(file.getAbsoluteFile());

BufferedWriter bw=**new** BufferedWriter(fw);

bw.write(article.toString());

bw.close();

System.*out*.println("Done writing article to file");

}**catch**(Exception ex){

ex.printStackTrace();

}

}

**Parsing**

**private** **void** parse() {

**try** {

JSONObject response = json.getJSONObject("response");

**if** (!response.isNull("docs")) {

resultsJSON = response.getJSONArray("docs");

searchResultList = **new** JSONFactoryImpl().searchResultListFromJSON(resultsJSON);

}

}**catch** (JSONException e) {

e.printStackTrace();

}

}

**Parsing each JSON field**

**private** **void** parse() **throws** JSONException {

searchResultsJSONList = **new** ArrayList<JSONObject>();

searchResults = **new** ArrayList<SearchResult>();

articles = **new** ArrayList<Article>();

**for** (**int** i = 0; i < json.length(); i++) {

JSONObject ob = json.getJSONObject(i);

searchResultsJSONList.add(ob);

JSONFactoryImpl factory = **new** JSONFactoryImpl();

//searchResults.add(factory.searchResultFromJSON(ob));

**if**("article".equalsIgnoreCase((String)ob.get("document\_type")))

articles.add(factory.articleFromJSON(ob));

}

}

The method writeArticleToFile writes the articles to files corresponding to the directory of their publish dates. We can use the hdfs –put command to put these files to HDFS.

**Pre-Processing the Data**  
In order to preprocess the date, we shall use only the limited fields required and we can filter that by passing it as an fl parameter to the API.

Following are the fields we need**:**document\_type,headline,news\_desk,pub\_date,source,web\_url,snippet,lead\_paragraph,abstract,blog

Following is a sample request-response for the same:

Request URI:  
http://api.nytimes.com/svc/search/v2/articlesearch.json?q=walmart&begin\_date=20150101&end\_date=20151031&fl=document\_type%2Cheadline%2Cnews\_desk%2Cpub\_date%2Csource%2Cweb\_url%2Csnippet%2Clead\_paragraph%2Cabstract%2Cblog&api-key=sample-key

Response:

{   
            "document\_type": "article",   
            "headline": {   
                "main": "Walmart Stock Sinks After a Warning on Sales",   
                "print\_headline": "Walmart Cuts Back Its Forecast "   
            },   
            "news\_desk": "Business",   
            "pub\_date": "2015-10-15T00:00:00Z",   
            "source": "The New York Times",   
            "web\_url":"http:\/\/www.nytimes.com\/2015\/10\/15\/business\/walmart-sales-forecast-share-buyback.html",   
            "snippet": "The retailer also said its heavy investment in wages and e-commerce would weigh on future earnings.",   
            "lead\_paragraph": "The retailer also said its heavy investment in wages and e-commerce would weigh on future earnings.",   
            "abstract": "Walmart upsets investors by saying sales will be flat in 2015; shares fall 10.04 percent to $60.03, largest drop in 17 years; cites big investments in wages and e-commerce competition as factors.",   
            "blog": []   
        }

**Google News API**

**Identify source of data**The Google News Search API has been officially deprecated as of May 26, 2011. The Google News Search API provides a JavaScript interface to embed Google News Search results in your website or application.

Instead of going on with this option, Google Api was very useful to display raw format of Json response with which we could search on the basis of terms as se search normally on google with specifies date range which is mentioned in Julian Format.

Sample google query will look something like this:

[https://ajax.googleapis.com/ajax/services/search/web?q=sony+daterange%3A2456931-2457022&start=0&rsz=8&v=1.0](https://webmail.utdallas.edu/owa/redir.aspx?SURL=rA0hL28QMPsgoM4-xQfHrGVxeYFp388D93aK84jiC3iRuPww3ePSCGgAdAB0AHAAcwA6AC8ALwBhAGoAYQB4AC4AZwBvAG8AZwBsAGUAYQBwAGkAcwAuAGMAbwBtAC8AYQBqAGEAeAAvAHMAZQByAHYAaQBjAGUAcwAvAHMAZQBhAHIAYwBoAC8AdwBlAGIAPwBxAD0AcwBvAG4AeQArAGQAYQB0AGUAcgBhAG4AZwBlACUAMwBBADIANAA1ADYAOQAzADEALQAyADQANQA3ADAAMgAyACYAcwB0AGEAcgB0AD0AMAAmAHIAcwB6AD0AOAAmAHYAPQAxAC4AMAA.&URL=https%3a%2f%2fajax.googleapis.com%2fajax%2fservices%2fsearch%2fweb%3fq%3dsony%2bdaterange%253A2456931-2457022%26start%3d0%26rsz%3d8%26v%3d1.0)

Here the term which we are looking for is “sony” with the given date range from 1st October to 31st December which is converted to Julian Format. The raw data file is attached named google\_json\_response.txt.

**Format of the data**The retuned JSON response contains following structure:

{

    "responseData": {

        "results": [{

            "GsearchResultClass": "GwebSearch",

            "unescapedUrl":

            "url":

            "visibleUrl":

            "cacheUrl":

            "title":

            "titleNoFormatting":

            "content":

        },{

        ...

        }

        ],

        "cursor": {

            "resultCount": "7,160,000",

            "pages": [{

                "start": "0",

                "label": 1

            },

            {(...page label attributes...)

            }

            ],

            "estimatedResultCount":

            "currentPageIndex":

            "moreResultsUrl":

            "searchResultTime":

        }

    },

    "responseDetails": null,

    "responseStatus": 200

}

**Accessing the data**With the following JSON response we can extract “unescapedUrl” further those webpages can be crawled to extract data. To be able to crawl different blog post we can extract “unescapedUrl” with the following PHP script.

<?php

$url = "https://ajax.googleapis.com/ajax/services/search/web?q=sony+daterange%3A2456931-2457022&start=0&rsz=8&v=1.0";

$ch = curl\_init();

curl\_setopt($ch, CURLOPT\_URL, $url);

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, 1);

curl\_setopt($ch, CURLOPT\_REFERER, "www.google.com");

$body = curl\_exec($ch);

curl\_close($ch);

$json = json\_decode($body);

echo '<br>';

for( $i=0;$i<4;$i = $i + 1){

print\_r( $json->responseData->results[$i]->unescapedUrl);

echo '<br>';

}

?>

The output of the above script will be list of URLs which are needed to be crawled in order to extract data from the same.

**Parsing the Data**A webcrawler needs to be designed which will crawl over the URL’s obtained from above script and depending on the type of data (json or xml); a parser will parse it.

**Pre-processing the Data**The data parsed from the webpage may contain irrelevant textual data which is to removed. Only relevant data is to be saved and used for further processing. Relevant data include words from titles, abstract, body and titles of relevant articles.

1. **MapReduce Solution**

**Investigating how Map Reduce will be applied to the problem**

The input for this project is the articles or blog posts about a company in a specific range of time. The input for each mapper will be a document (article or blog post) along with the date of that document being posted online.

We will use Map Reduce to process the document and determine if it is related to cyber security. This will be determined by taking a count of all the cyber security related terms in the document and dividing it by the total no. of unique words in the document. Thus, we will obtain a ratio of the cyber security related terms. If this ratio is higher than a certain threshold, the document would be classified as cyber security related. If the ratio is less than the threshold it will be classified as not related to cyber security. We will ignore the stop words and punctuations for this task to increase accuracy.

To get the threshold we will first run a script on documents which we know are cyber security related. This script will calculate the ratio of the cyber security terms to the total no. of unique words. We will then take an average of this ratio over several documents and that average will be our threshold T.

**Identifying input, intermediate, and output key-value pairs**

The input for this project is the articles or blog posts about a company in a specific range of time.

The input for each mapper will be a document (article or blog post) along with the date of that document being posted online.

In terms of **key value pairs**:

Input for the mapper will be:

Key: Document Id (Document name and Date posted)

Value: Content of the document

Output of the mapper:

Key: Document Id (Document name and Date posted)

Value: Cyber Security term found in the document

Input for the reducer:

Key: Document Id (Document name and Date posted)

Value: List of Cyber Security term found in the document

Output of the reducer:

Key: Document Id (Document name and Date posted)

Value: Boolean indicating if a document is cyber security related or not

**Identifying what type of combiner and custom partitioner will be needed**

We will not need any custom partitioner or combiner.

**Identifying output of Map and input/output of Reduce phase**

Output of the mapper:

Key: Document Id (Document name and Date posted)

Value: Cyber Security term found in the document

Input for the reducer:

Key: Document Id (Document name and Date posted)

Value: List of Cyber Security term found in the document

Output of the reducer:

Key: Document Id (Document name and Date posted)

Value: Boolean indicating if a document is cyber security related or not

1. **Validation Strategy**

**Classification Algorithm**

A classification algorithm will be used to classify each article as cyber-security related or not based on a threshold value which will be generated by training.

We will manually tag articles as cyber security related or not for about 50 articles and we will calculate the threshold value from these articles. Once we have the threshold value, we will use it classify unknown articles.

**Output Validation**

We will take around 50 articles and manually tag them as cyber security related or not. We will then divide these 50 articles into training and testing se. The split will be 90:10 where 90% articles will be used for training and 10% will be used for testing the accuracy of our classifier.

If we find that the accuracy of our classifier is too low, we will increase the training set data.

We will then use the learned classifier to determine if an article is cyber security related or not.

**Accuracy of the method**

We will run our classifier on the testing data set and we will calculate the accuracy as follows:

Accuracy of classifier = No. of articles classified correctly / Total no. of articles

% accuracy = Accuracy of Classifier \* 100

**Advantages and disadvantages of our method**

**Advantages:**

The classifier is simple to implement.

It will require very less training data to learn.

Training time will be very less.

**Disadvantages:**

If the amount of training data is too high, it may lead to over fitting.