Wpływ aktywności na Twitterze na notowania spółek giełdowych

Raport z projektu

Zespół *Chmurki*
Sebastian Deręgowski, Klaudia Gruszkowska, Bartosz Jamroży

Spis treści

1. Wstęp

Nasz projekt skupia się na analizie wpływu postów z serwisu społecznościowego Twitter na ceny akcji największych, światowych firm. W poniższym dokumencie opiszemy źródła danych, wykorzystany przez nas stos architektoniczny oraz testy.

2. Opis danych

Pierwszy zbiór danych dotyczy cen akcji kilku spółek notowanych na nowojorskiej giełdzie papierów wartościowych (NYCE). Analizujemy je pod kątem zmieniania się notowań spółek: Google'a, Microsoftu, Apple'a oraz Tesli w interwałach pięciominutowych. Dane pobieramy przy pomocy API ze strony www.alphavantage.co.

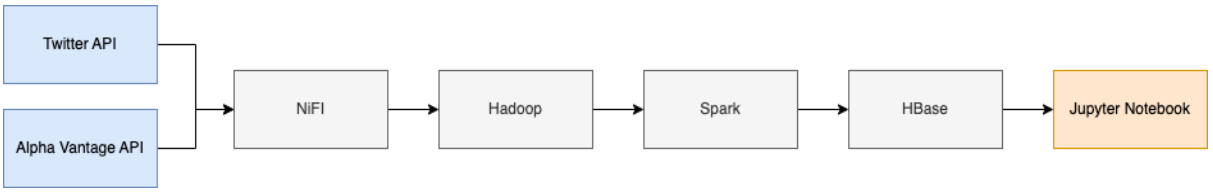
Dane są zwracane przez API w formacie CSV i zawierają informacje nt. kursu otwarcia, zamknięcia, a także najmniejszej i największej wartości w danym przedziale czasowym (przykład poniżej).

	A	B	C	D	E	F
1	timestamp	open	high	low	close	volume
2	1/6/2023 20:00	224.96	225.05	224.96	225.04	3006
3	1/6/2023 19:55	224.95	225	224.94	224.95	3055
4	1/6/2023 19:50	224.91	224.91	224.91	224.91	135
5	1/6/2023 19:45	224.9	224.93	224.9	224.93	666
6	1/6/2023 19:30	224.95	224.95	224.95	224.95	325
7	1/6/2023 19:25	225	225.05	225	225.05	253
8	1/6/2023 19:20	225	225	225	225	205
9	1/6/2023 19:15	225.0499	225.0499	225.0499	225.0499	495
10	1/6/2023 19:05	225.05	225.05	225.05	225.05	454
11	1/6/2023 19:00	225	225	225	225	511
12	1/6/2023 18:55	225.04	225.05	225.04	225.05	935
13	1/6/2023 18:30	225.04	225.04	225.04	225.04	977

Drugim zbiorem danych jest Twittera. A dokładniej dane pobierane przez API z serwisu Twitter dla developreów (https://developer.twitter.com/en/). Do każdej z analizowanych spółek zbieramy dane z konkretnych hashtagów powiązanych z tymi spółkami. Dane są zwracane w postaci JSONa zawierającego informacje dotyczące początku i końca rozpatrywanego okresu i ilości tweetów zawierających dany hashtag. Rozpatrujemy okresy czasowe jednominutowe i hashtagi: #Apple. #Google, #Microsoft, #Tesla. Poniżej screen z przykładowej odpowiedzi Twittera.

```
{
    "data": [
        {
            "end": "2023-01-05T09:16:00.000Z",
            "start": "2023-01-05T09:15:44.000Z",
            "tweet_count": 2
        },
        {
            "end": "2023-01-05T09:17:00.000Z",
            "start": "2023-01-05T09:16:00.000Z",
            "tweet_count": 4
        },
        {
            "end": "2023-01-05T09:18:00.000Z",
            "start": "2023-01-05T09:17:00.000Z",
            "tweet_count": 2
        },
        {
            "end": "2023-01-05T09:19:00.000Z",
            "start": "2023-01-05T09:18:00.000Z",
            "tweet_count": 1
        },
        {
            "end": "2023-01-05T09:20:00.000Z",
            "start": "2023-01-05T09:19:00.000Z",
            "tweet_count": 4
        },
```


3. Stos architektoniczny



Dane pobieramy z API źródeł opisanych w rozdziale 2 za pomocą Apache NiFi. Następnie również w Apache NiFi transformujemy pliki, tak aby uzyskać złączone pliki w formacie parquet na Apache Hadoop. Każdy plik twittera zawiera dane z jednego dnia i dotyczy jednego z wyszukiwanych hashtagów. Pliki alphaventage dotyczą jednej z obserwowanych spółek i zawierają historię z kilkunastu ostatnich dni.

W Apache Hadoop pliki pogrupowane są w dwóch katalogach zależnie od źródła danych.

Następnym krokiem naszego przepływu danych jest wykorzystanie narzędzia Apache Spark. W tym miejscu wgrywane pliki z Apache Hadoop filtrujemy tak aby pokazywały tylko interesujące nas zakresy czasowe. Łączymy dane z różnych plików po wartości ‘timestamp’ i zapisujemy je do dwóch tabel, jednej zbierającej wszystkie informacje twitterowe i drugiej zbierającej dane alphaventage.

Dane przetworzone przez Apache Spark lądują w tabelach stworzonych w Apache HBase. Później wykorzystujemy je do analizy za pomocą narzędzia Jupyter Notebook. Jupyter Notebook ma za zadanie symulację narzędzia typu Business Intelligence umożliwiając użytkownikowi końcowemu dostęp do wybranych statystyk i analiz. Użytkownik ma możliwość modyfikowania parametrów funkcji takich jak przedział czasowy, kurs otwarcia zamknięcia, czy firmy brane pod uwagę.

Funkcja *volume_stats_compare()* pozwala na porównanie wolumenów dwóch wybranych firm w wybranym przedziale czasowym.

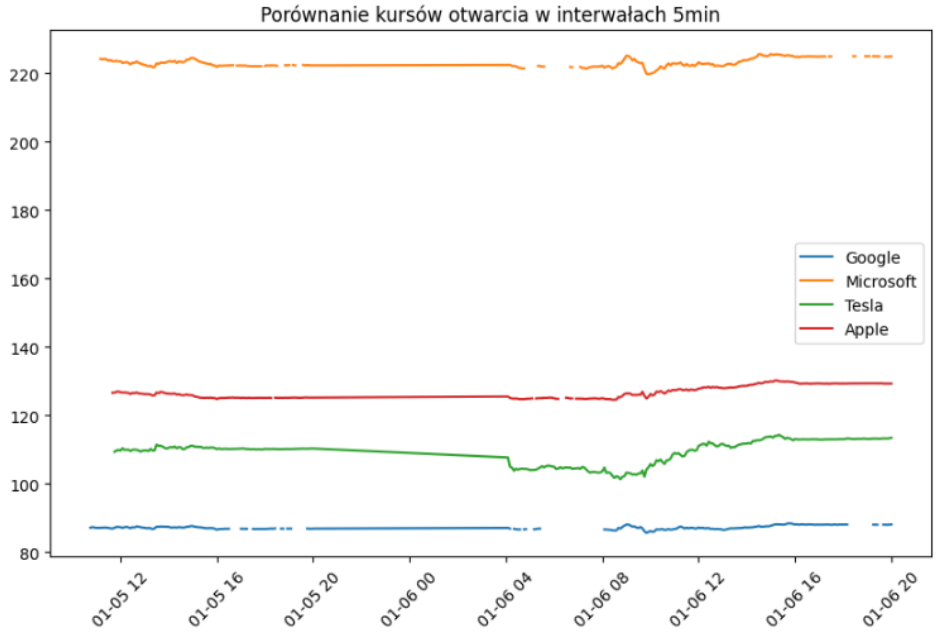
```
In [5]: volume_stats_compare()
```

```
Porównanie wolumenów dla firm Tesla oraz Google w zadanym okresie:

  Statystyka         Tesla        Google       Różnica
0       mean  1.019221e+06  1.503219e+05  8.688987e+05
1        std  1.352177e+06  1.832788e+05  1.168899e+06
2        min  3.569000e+03  1.000000e+02  3.469000e+03
3        max  9.783541e+06  1.537600e+06  8.245941e+06
```

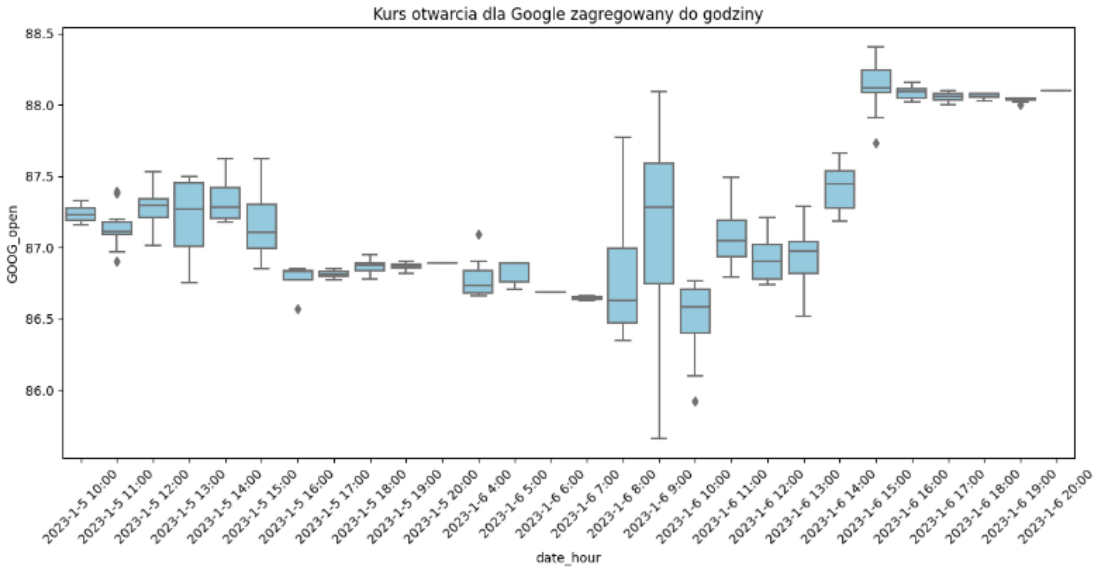
Funkcja *plot_companies()* umożliwia porównanie kursów otwarcia lub zamknięcia wybranych film w wybranym przedziale czasowym.

```
In [6]: plot_companies()
```



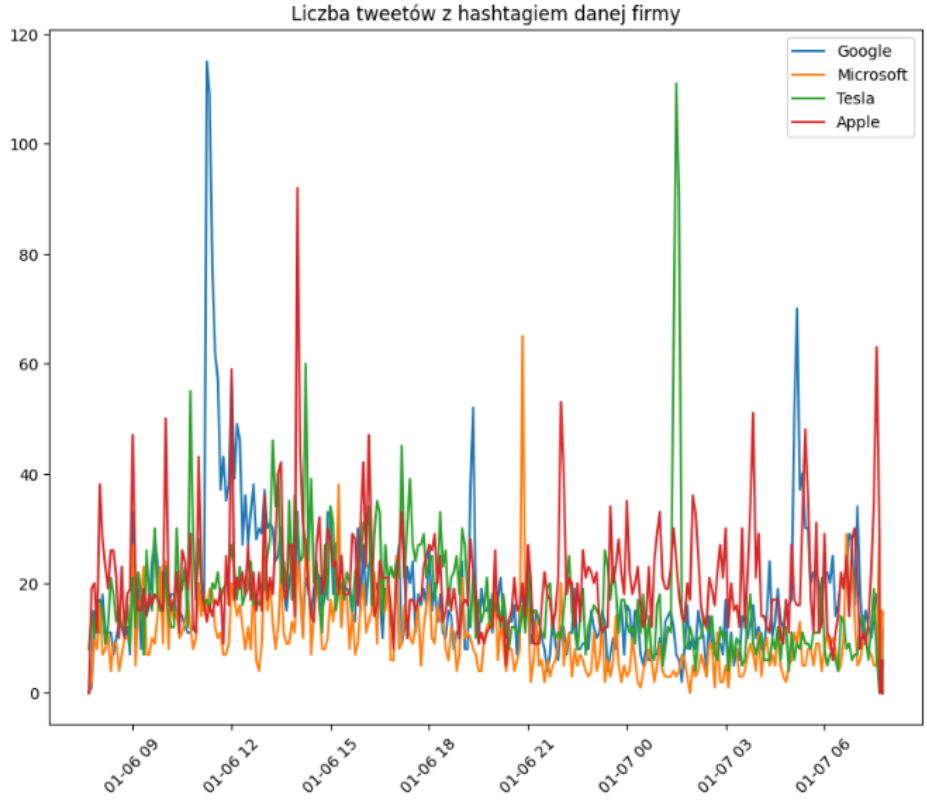
Funkcja *plot_boxplot()* umożliwia szczegółową analizę kursu otwarcia bądź zamknięcia dla wybranej firmy z danymi zagregowanymi do godziny.

```
In [7]: plot_boxplot()
```



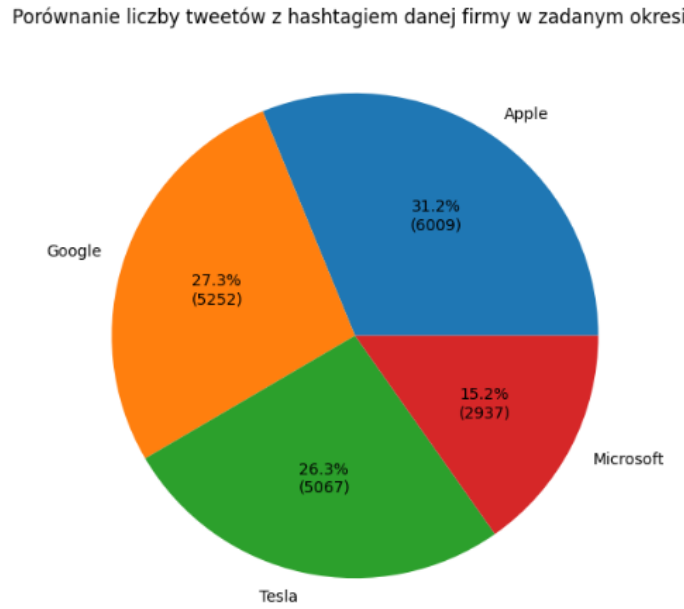
Funkcja *plot_tweets()* umożliwia porównanie liczby tweetów z hashtagiem wybranych firm w wybranym przedziale czasowym. Istnieje możliwość wyboru między wykresem liniowym a skumulowaną linią trendu.

```
In [8]: plot_tweets()
```



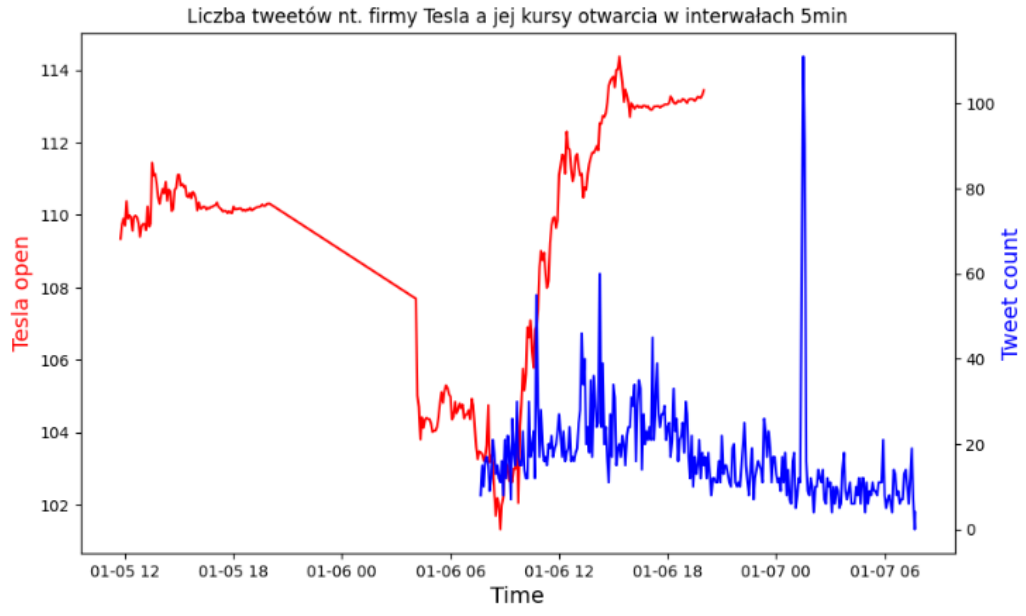
Funkcja *tweets_pie()* porównuje liczbę tweetów na wykresie kołowym w wybranym przedziale czasowym.

```
In [9]: tweets_pie()
```



Funkcja *tweets_stock()* zestawia ze sobą liczbę tweetów nt. firmy Tesla z jej kolejnymi kursami otwarcia w interwałach pięciominutowych.

```
In [10]: tweets_stock()
```



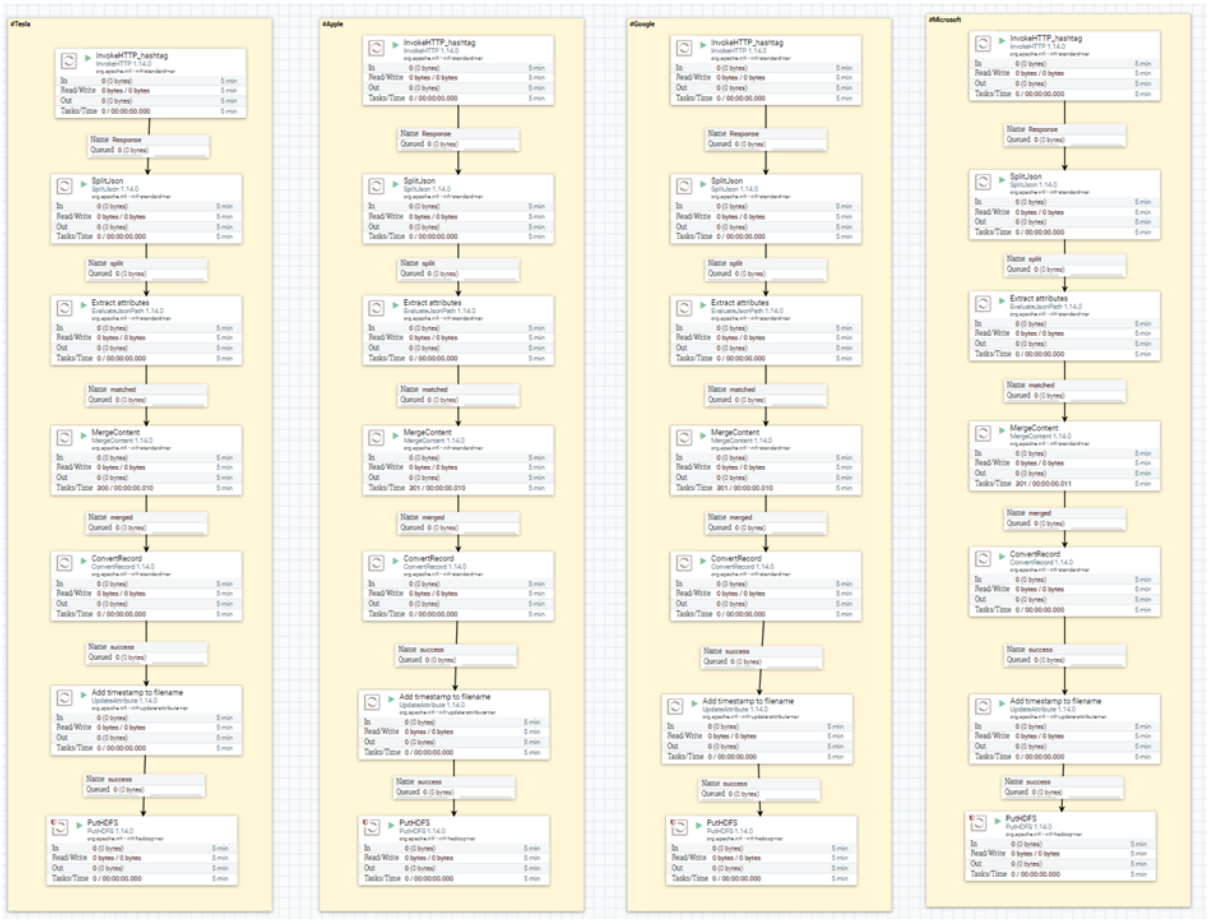
4. Testy


```
{
    "Meta Data": {
        "1. Information": "Intraday (5min) open, high, low, close prices and volume",
        "2. Symbol": "TSLA",
        "3. Last Refreshed": "2023-01-06 20:00:00",
        "4. Interval": "5min",
        "5. Output Size": "Full size",
        "6. Time Zone": "US/Eastern"
    },
    "Time Series (5min)": {
        "2023-01-06 20:00:00": {
            "1. open": "113.4401",
            "2. high": "113.7500",
            "3. low": "113.4400",
            "4. close": "113.6800",
            "5. volume": "71404"
        },
        "2023-01-06 19:55:00": {
            "1. open": "113.3100",
            "2. high": "113.4600",
            "3. low": "113.3000",
            "4. close": "113.4500",
            "5. volume": "29992"
        },
        "2023-01-06 19:50:00": {
            "1. open": "113.2300",
            "2. high": "113.3100",
            "3. low": "113.2300",
            "4. close": "113.3100",
            "5. volume": "19735"
        },
        "2023-01-06 19:45:00": {
            "1. open": "113.2400",
            "2. high": "113.2500",
            "3. low": "113.2100",
            "4. close": "113.2300",
            "5. volume": "11671"
        },
        "2023-01-06 19:40:00": {
            "1. open": "113.2600",
            "2. high": "113.3000",
            "3. low": "113.2499",
            "4. close": "113.2700",
            "5. volume": "13708"
        },
        "2023-01-06 19:35:00": {
            "1. open": "113.2000",
            "2. high": "113.2700",
            "3. low": "113.1900",
            "4. close": "113.2600",
            "5. volume": "18536"
        },
        "2023-01-06 19:30:00": {
            "1. open": "113.1400",
            "2. high": "113.2000",
            "3. low": "113.1400",
            "4. close": "113.1900",
            "5. volume": "6636"
        },
        "2023-01-06 19:25:00": {
            "1. open": "113.1900",
            "2. high": "113.2000",
            "3. low": "113.1000",
            "4. close": "113.1400",
            "5. volume": "9779"
```

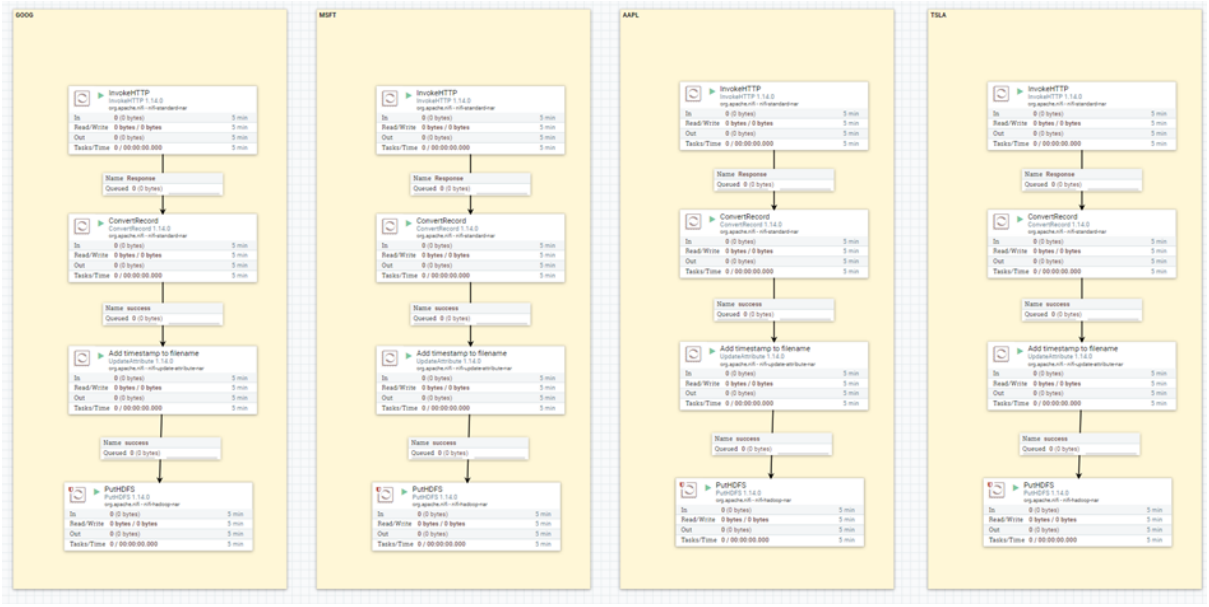

Test poprawności działania API giełdowego, api poprawnie odpowiada na zapytanie zwracając informacje na temat kursu akcji w danych momentach czasowych.

```
StatusCode        : 200
StatusDescription : OK
Content           : {"data":[{"end":"2023-01-05T00:01:00.000Z","start":"2023-01-05T00:00:00.000Z","tweet_count":6},{"end":"2023-01-05T00:02:00.000Z","start":"2023-01-05T00:01:00.000Z","tweet_count":5},{"end":"2023-01-05T...
RawContent        : HTTP/1.1 200 OK
                    perf: 7626143928
                    api-version: 2.60
                    x-access-level: read
                    x-frame-options: SAMEORIGIN
                    x-transaction-id: e190500fbf3d4d2d
                    x-xss-protection: 0
                    x-rate-limit-limit: 300
                    x-rate-limit-...
Forms             : {}
Headers           : {[perf, 7626143928], [api-version, 2.60], [x-access-level, read], [x-frame-options, SAMEORIGIN]...}
Images            : {}
InputFields       : {}
Links             : {}
ParsedHtml        : mshtml.HTMLDocumentClass
RawContentLength  : 123909
```

Test poprawności API twitterowego, interfejs poprawie zwraca informację w formacie Json o liczbie tweetów dla danego hasztaga.



Przetwarzanie danych o tweetach. Wizualna kontrola stanu procesorów w narzędziu nifi. Uruchomione oraz zaplanowane procesory nie raportują błędów.



Przetwarzanie danych o notowaniach giełdowych. Wizualna kontrola stanu procesorów w narzędziu nifi. Uruchomione oraz zaplanowane procesory nie raportują błędów.

```
vagrant@node1:~$ hadoop fs -ls /user/project/twitter
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/apache-tez-0.9.1-bin/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Found 12 items
-rw-r--r--   1 root supergroup      26035 2023-01-07 09:03 /user/project/twitter/Apple_Hashtag_2023-01-06-221311129Z.parquet
-rw-r--r--   1 root supergroup      57837 2023-01-07 08:45 /user/project/twitter/Apple_Hashtag_2023-01-07-084506462Z.parquet
-rw-r--r--   1 root supergroup      26163 2023-01-07 08:45 /user/project/twitter/Apple_Hashtag_2023-01-07-084506486Z.parquet
-rw-r--r--   1 root supergroup      26020 2023-01-07 09:05 /user/project/twitter/Google_Hashtag_2023-01-06-221311129Z.parquet
-rw-r--r--   1 root supergroup      57841 2023-01-07 08:47 /user/project/twitter/Google_Hashtag_2023-01-07-084736542Z.parquet
-rw-r--r--   1 root supergroup      26158 2023-01-07 08:47 /user/project/twitter/Google_Hashtag_2023-01-07-084736557Z.parquet
-rw-r--r--   1 root supergroup      57676 2023-01-07 09:05 /user/project/twitter/Microsoft_Hashtag_2023-01-06-221311129Z.parquet
-rw-r--r--   1 root supergroup      57666 2023-01-07 08:48 /user/project/twitter/Microsoft_Hashtag_2023-01-07-084842432Z.parquet
-rw-r--r--   1 root supergroup      26073 2023-01-07 08:48 /user/project/twitter/Microsoft_Hashtag_2023-01-07-084842433Z.parquet
-rw-r--r--   1 root supergroup      23253 2023-01-07 09:03 /user/project/twitter/Tesla_Hashtag_2023-01-06-221311129Z.parquet
-rw-r--r--   1 root supergroup      57813 2023-01-07 08:40 /user/project/twitter/Tesla_Hashtag_2023-01-07-084052090Z.parquet
-rw-r--r--   1 root supergroup      26163 2023-01-07 08:40 /user/project/twitter/Tesla_Hashtag_2023-01-07-084052113Z.parquet
```

Test zapisu plików przez nifi do systemu plików w hadoop. Proces odpowiedzialny za informacje o tweetach poprawnie umieszcza pliki w hdfs.

```
vagrant@node1:~$ hadoop fs -ls /user/project/alphavantage
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/apache-tez-0.9.1-bin/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Found 8 items
-rw-r--r--   1 root supergroup       5798 2023-01-06 22:13 /user/project/alphavantage/AAPL_2023-01-06-221319990Z.parquet
-rw-r--r--   1 root supergroup     148780 2023-01-07 09:56 /user/project/alphavantage/AAPL_2023-01-07-095651640Z.parquet
-rw-r--r--   1 root supergroup       5787 2023-01-06 22:01 /user/project/alphavantage/GOOG_2023-01-06-220108467Z.parquet
-rw-r--r--   1 root supergroup     111144 2023-01-07 09:56 /user/project/alphavantage/GOOG_2023-01-07-095648729Z.parquet
-rw-r--r--   1 root supergroup       5901 2023-01-06 22:05 /user/project/alphavantage/MSFT_2023-01-06-220538635Z.parquet
-rw-r--r--   1 root supergroup     133360 2023-01-07 09:56 /user/project/alphavantage/MSFT_2023-01-07-095649876Z.parquet
-rw-r--r--   1 root supergroup       5878 2023-01-06 22:13 /user/project/alphavantage/TSLA_2023-01-06-221311129Z.parquet
-rw-r--r--   1 root supergroup     158444 2023-01-07 09:56 /user/project/alphavantage/TSLA_2023-01-07-095654496Z.parquet
```

Test zapisu plików przez nifi do systemu plików w hadoop. Proces odpowiedzialny za informacje o notowaniach spółek poprawnie umieszcza pliki w hdfs.

```
>>> TSLA=spark.read.parquet('hdfs://localhost:8020//user/project/alphavantage/TSLA_2023-01-07-095654496Z.parquet')
>>> TSLA.show()
+-------------------+--------+------+--------+------+------+
|          timestamp|    open|  high|     low| close|volume|
+-------------------+--------+------+--------+------+------+
|2023-01-06 20:00:00|113.4401|113.75|  113.44|113.68| 71404|
|2023-01-06 19:55:00|  113.31|113.46|   113.3|113.45| 29992|
|2023-01-06 19:50:00|  113.23|113.31|  113.23|113.31| 19735|
|2023-01-06 19:45:00|  113.24|113.25|  113.21|113.23| 11671|
|2023-01-06 19:40:00|  113.26| 113.3|113.2499|113.27| 13708|
|2023-01-06 19:35:00|   113.2|113.27|  113.19|113.26| 18536|
|2023-01-06 19:30:00|  113.14| 113.2|  113.14|113.19|  6636|
|2023-01-06 19:25:00|  113.19| 113.2|   113.1|113.14|  9779|
|2023-01-06 19:20:00|   113.2| 113.2|  113.15| 113.2|  7530|
|2023-01-06 19:15:00|   113.2|113.21|  113.17|113.18|  6620|
|2023-01-06 19:10:00|  113.18| 113.2|  113.18|113.18|  5292|
|2023-01-06 19:05:00|  113.09|113.19|  113.09|113.17| 10241|
|2023-01-06 19:00:00|  113.13|113.14|  113.07| 113.1| 23811|
|2023-01-06 18:55:00|  113.18|113.18|  113.12|113.12| 13814|
|2023-01-06 18:50:00|   113.2| 113.2|  113.17|113.18|  9488|
|2023-01-06 18:45:00|  113.13| 113.2|  113.13| 113.2| 13990|
|2023-01-06 18:40:00|  113.13|113.17|  113.09|113.13| 19582|
|2023-01-06 18:35:00|113.1499|113.18|  113.12|113.14|  5860|
|2023-01-06 18:30:00|113.0899|113.15|  113.08|113.14| 12493|
|2023-01-06 18:25:00|  113.07|113.12|  113.06|113.08| 13162|
+-------------------+--------+------+--------+------+------+
only showing top 20 rows
```

Sprawdzenie czy plik parquet stworzony przez nifi poprawnie przechowuje dane. Dane giełdowe udaje się poprawnie wyświetlić, plik zawiera oczekiwane wartości.

```
>>> TSLA_Hash=spark.read.parquet('hdfs://localhost:8020//user/project/twitter/Tesla_Hashtag_2023-01-07-084052113Z.parquet')
>>> TSLA_Hash.show()
+--------------------+--------------------+-----------+
|                 end|               start|tweet_count|
+--------------------+--------------------+-----------+
|2023-01-07T00:21:...|2023-01-07T00:20:...|          2|
|2023-01-07T00:22:...|2023-01-07T00:21:...|          3|
|2023-01-07T00:23:...|2023-01-07T00:22:...|          3|
|2023-01-07T00:24:...|2023-01-07T00:23:...|          3|
|2023-01-07T00:25:...|2023-01-07T00:24:...|          3|
|2023-01-07T00:26:...|2023-01-07T00:25:...|          6|
|2023-01-07T00:27:...|2023-01-07T00:26:...|          1|
|2023-01-07T00:28:...|2023-01-07T00:27:...|          1|
|2023-01-07T00:29:...|2023-01-07T00:28:...|          0|
|2023-01-07T00:30:...|2023-01-07T00:29:...|          4|
|2023-01-07T00:31:...|2023-01-07T00:30:...|          7|
|2023-01-07T00:32:...|2023-01-07T00:31:...|          4|
|2023-01-07T00:33:...|2023-01-07T00:32:...|          3|
|2023-01-07T00:34:...|2023-01-07T00:33:...|          0|
|2023-01-07T00:35:...|2023-01-07T00:34:...|          0|
|2023-01-07T00:36:...|2023-01-07T00:35:...|          3|
|2023-01-07T00:37:...|2023-01-07T00:36:...|          0|
|2023-01-07T00:38:...|2023-01-07T00:37:...|          4|
|2023-01-07T00:39:...|2023-01-07T00:38:...|          1|
|2023-01-07T00:40:...|2023-01-07T00:39:...|          5|
+--------------------+--------------------+-----------+
only showing top 20 rows
```

W przypadku danych o tweetach plik parquet również poprawnie przechowuje dane.

```
In 30  1 row = table.row(b'2023-01-05 16:20:00')
       2 print(row)

  {b'AAPL:close': b'125.13999938964844', b'AAPL:high': b'125.1500015258789', b'AAPL:low': b'125.08999633789062', b'AAPL:open': b'125.08999633789062', b'AAPL:volume': b'23147',
   b'GOOG:close': b'86.80000305175781', b'GOOG:high': b'86.8499984741211', b'GOOG:low': b'86.80000305175781', b'GOOG:open': b'86.8499984741211', b'GOOG:volume': b'1789', b'Id:Time':
   b'2023-01-05 16:20:00', b'MSFT:close': b'222.25999450683594', b'MSFT:high': b'222.3000030517578', b'MSFT:low': b'222.25999450683594', b'MSFT:open': b'222.3000030517578', b'MSFT:volume':
    b'849', b'TSLA:close': b'110.2300033569336', b'TSLA:high': b'110.25', b'TSLA:low': b'110.19999694824219', b'TSLA:open': b'110.2300033569336', b'TSLA:volume': b'18061'}
```

```
In 32  1 row = table.row(b'2023-01-06 16:20:00')
       2 print(row)

  {b'Hashtags:AAPL_tweet_count': b'14', b'Hashtags:GOOG_tweet_count': b'18', b'Hashtags:MSFT_tweet_count': b'22', b'Hashtags:TSLA_tweet_count': b'29', b'Id:Time': b'2023-01-06 16:20:00'}
```

Powyższe dwa zrzuty ekranu zostały wykonane w cele wybiórczego sprawdzenia czy skrypt sparkowy (pySpark) popranie złączył i załadował pliki do tabeli hBase. Dane wyświetlają się poprawnie dla obu dostępnych tabel w hBase.

Z poziomu JupyterNotebooka możemy sprawdzać statystyki danych zarówno twitterowych jak i z giełdy.

5. Podsumowanie

Wszystkie założenia projektu zostały spełnione. Dane z obu API są pobierane, przechwytywane, przetwarzane i składowane w określonej formie. Użytkownik końcowy z poziomu Jupyter Notebooka ma możliwość samodzielnej analizy danych.

Podczas pracy nad projektem niejednokrotnie spotkaliśmy się z różnorodnymi problemami. Konfiguracja usług była nieintuicyjna, dodatkowo pojawiały się kłopoty z instalowaniem bibliotek Pythonowych na maszynie wirtualnej, a same dane ze względu na swój format wymagały bardzo dużej i szczegółowej obróbki przed ich docelowym zapisaniem. Mimo to, udało nam się stworzyć rozwiązanie end-to-end, które uważamy za satysfakcjonujące.