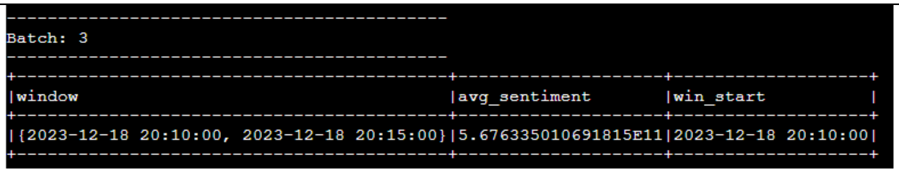
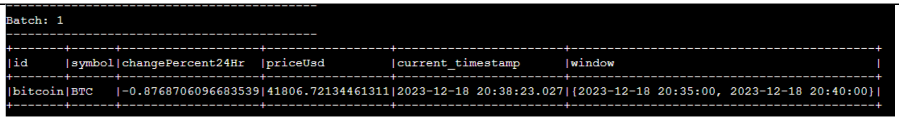
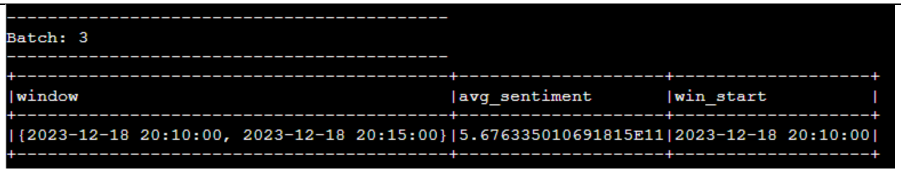
Tests Description

Test Name	Objective	Steps	Expected Results	Actual results
Sentiment analysis	Check if sentiment analysis works correctly.	1. Ensure that subscribed topic in kafka contains messages with articles 2. Run spark code with sentiment analysis code	New columns with with averaged sentiment in table is created	
Streaming rates	Check if streaming data from kafka to spark works correctly	1. Ensure that NiFI processors from crypto_rates_api_to_kafka are running 2. Run script test_streaming_rate.py, which is a copy of our solution that prints stream to terminal instead of joining data and dropping	Within 30 seconds after running the program spark printed table with columns: id, symbol, changePercent24Hr, and priceUSD. This table also can't be empty.	

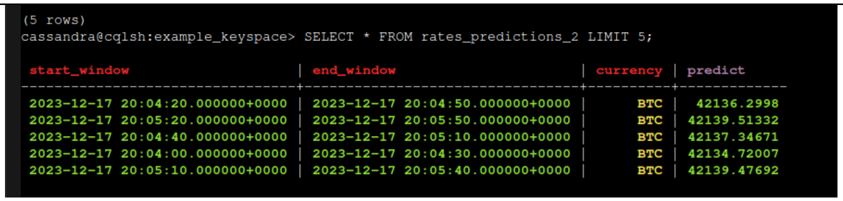
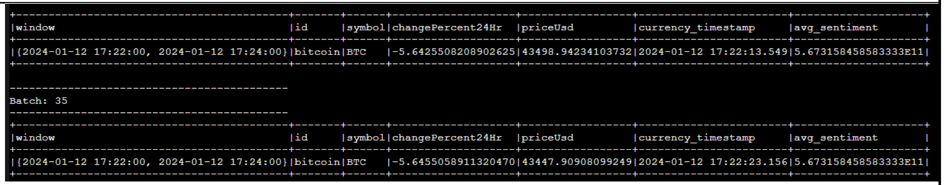
```
-------------------------------------------
Batch: 3
-------------------------------------------
+------------------------------------------+--------------------+-------------------+
|window                                    |avg_sentiment       |win_start          |
+------------------------------------------+--------------------+-------------------+
|{2023-12-18 20:10:00, 2023-12-18 20:15:00}|5.676335010691815E11|2023-12-18 20:10:00|
+------------------------------------------+--------------------+-------------------+
```

```
-------------------------------------------
Batch: 1
-------------------------------------------
+-------+------+-------------------+-----------------+-----------------------+------------------------------------------+
|id     |symbol|changePercent24Hr  |priceUsd         |current_timestamp      |window                                    |
+-------+------+-------------------+-----------------+-----------------------+------------------------------------------+
|bitcoin|BTC   |-0.8768706096683539|41806.72134461311|2023-12-18 20:38:23.027|{2023-12-18 20:35:00, 2023-12-18 20:40:00}|
+-------+------+-------------------+-----------------+-----------------------+------------------------------------------+
```

		it to cassandra 3. Ensure that within 30 seconds after running the program spark printed table with columns: id, symbol, changePercent24Hr, and priceUSD. Also ensure that this table isn’t empty.		
Streaming rates	Check if streaming data from kafka to spark works correctly	1. Ensure that NiFI processors from NewsAPI_news_api_to_kafka are running 2. Run script simulate_news.py, which takes newest article from	Within 15 minutes after running the program spark printed non empty table with expected columns	

```
-------------------------------------------
Batch: 3
-------------------------------------------
+------------------------------------------+-------------------+-------------------+
|window                                    |avg_sentiment      |win_start          |
+------------------------------------------+-------------------+-------------------+
|{2023-12-18 20:10:00, 2023-12-18 20:15:00}|5.676335010691815E11|2023-12-18 20:10:00|
+------------------------------------------+-------------------+-------------------+
```

		kafka topic waits 30 seconds and puts it back simulating data ingestion 3. Run script test_streaming_news.py, which is a copy of our solution that prints stream to terminal instead of joining data and dropping it data to cassandra 4. Ensure that within 15 minutes after running the program spark printed table with columns: window, sentiment. Also ensure		

		that this table isn't empty.		
Machine Learning predictions	Checking if models generate predictions and if they are saved to Cassandra	1. Ensure that data is retrieved from Kafka. 2. Ensure that models generate correct predictions (not a constant value). 3. Check if data is saved to Cassandra.	After a few minutes after running a script, the predictions should be available in Cassandra.	
Rates and sentiment join	Checking if joining streams of crytocurrency rates and aggregated sentiment from the articles on window works correctly.	1. Run the script for calculating sentiment from incoming news articles 2. Run the script for reading the crypto rates stream	Results of successful join are saved to kafka topic. (new column avg_sentiment and window in rates data)	

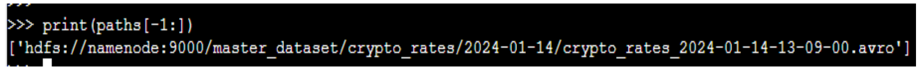
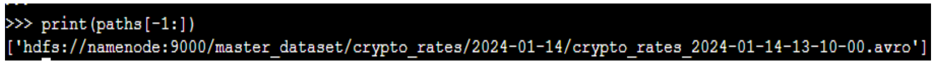
```
(5 rows)
cassandra@cqlsh:example_keyspace> SELECT * FROM rates_predictions_2 LIMIT 5;

 start_window                    | end_window                      | currency | predict
---------------------------------+---------------------------------+----------+-------------
 2023-12-17 20:04:20.000000+0000 | 2023-12-17 20:04:50.000000+0000 |      BTC |  42136.2998
 2023-12-17 20:05:20.000000+0000 | 2023-12-17 20:05:50.000000+0000 |      BTC | 42139.51332
 2023-12-17 20:04:40.000000+0000 | 2023-12-17 20:05:10.000000+0000 |      BTC | 42137.34671
 2023-12-17 20:04:00.000000+0000 | 2023-12-17 20:04:30.000000+0000 |      BTC | 42134.72007
 2023-12-17 20:05:10.000000+0000 | 2023-12-17 20:05:40.000000+0000 |      BTC | 42139.47692
```

```
+------------------------------------------+-------+------+-------------------+-------------------+--------------------+--------------------+
|window                                    |id     |symbol|changePercent24Hr  |priceUsd           |currency_timestamp  |avg_sentiment       |
+------------------------------------------+-------+------+-------------------+-------------------+--------------------+--------------------+
|{2024-01-12 17:22:00, 2024-01-12 17:24:00}|bitcoin|BTC   |-5.6425508208902625|43498.94234103732|2024-01-12 17:22:13.549|5.673158458583333E11|
+------------------------------------------+-------+------+-------------------+-------------------+--------------------+--------------------+

-------------------------------------------
Batch: 35
-------------------------------------------
+------------------------------------------+-------+------+-------------------+-------------------+--------------------+--------------------+
|window                                    |id     |symbol|changePercent24Hr  |priceUsd           |currency_timestamp  |avg_sentiment       |
+------------------------------------------+-------+------+-------------------+-------------------+--------------------+--------------------+
|{2024-01-12 17:22:00, 2024-01-12 17:24:00}|bitcoin|BTC   |-5.6455058911320470|43447.90908099249|2024-01-12 17:22:23.156|5.673158458583333E11|
+------------------------------------------+-------+------+-------------------+-------------------+--------------------+--------------------+
```

		3. Run the script for joining the stream ans saving it to kafka topic		

Raw ingestion	Checking if new currency rates are ingested into HDFS using NiFi and if API works correctly	1. Ensure that NiFi process group called crypto_rates_api_to_master is working
		2. Edit script test_raw_in_hdfs.py changing the date to current one
		3. Run script test_raw_in_hdfs.py it will return the name of the newest file in HDFS. Remember this name
		4. Run script again no sooner than 1 minute
			After running program again after at least 1 minute name of newest file will be different	Results after running program once
				
				`>>> print(paths[-1:])`
				`['hdfs://namenode:9000/master_dataset/crypto_rates/2024-01-14/crypto_rates_2024-01-14-13-09-00.avro']`
				Results after running program one minute later
				
				`>>> print(paths[-1:])`
				`['hdfs://namenode:9000/master_dataset/crypto_rates/2024-01-14/crypto_rates_2024-01-14-13-10-00.avro']`

		later. Check if the name of newest file has changed.		