Big Data report

team: Magiczny Krzysztof

Spaliński Krzysztof,
Kała Jakub,
Dyczko Michał,
Pawłowski Maciej

June 2020

1 Project description

Memelysis - Website for memes analysis, with respect to themes and a place to find your favourite meme. Data sources will consist of: Imgur, Reddit, Twitter, Memedroid.

1.1 Delivered product

Users are be provided with a web application consisting of three main components - a dashboard with a filtering tools and statistical analysis module.

Dashboard is presenting the latest data obtained from the most popular portals providing user-generated content. Each presented object will be extended with information such as source, matched category and virality value. Instead of looking through dozens of websites one will be able to enjoy all the memes in one place. Moreover each user will have its own personalised content thanks to the minimum virality level that can be set in the dashboard. With the filtering tools one will enable the client to extract memes on dedicated topics. In such a case the user will have the access to the most well-received content from the most trendy websites.

With the statistical analysis provided, users might examine current trends. They can track their favourite content and observe others' meme preferences. Such tools might broaden their taste in memes and invite them to a meme world they have not experienced before.

2 Data

2.1 Data Sources

2.1.1 9GAG

"9GAG is a Hong Kong–based online platform and social media website, which allows its users to upload and share "user-generated content" or other content from external social media websites. It was launched on July 1, 2008." [1]. Currently it is used by more than 200 million users a month.

[**Milestone 2**] We have decided not to use 9GAG as we would like to respect the source' scraping policy. Instead, we used imgur as a replacement data source.

2.1.2 Twitter

Twitter is an American microblogging and social networking service on which users post and interact with messages known as "tweets". Twitter was launched in July 2006. The service rapidly gained worldwide popularity. In 2012, more than 100 million users posted 340 million tweets a day. Users can group posts together by topic or type by use of hashtags – words

[1] https://en.wikipedia.org/wiki/9GAG

or phrases prefixed with a “#” sign. Tweets include also the functionality of attaching an image.

2.1.3 Reddit

”Reddit is an American social news aggregation, web content rating, and discussion website. Registered members submit content to the site such as links, text posts, and images, which are then voted up or down by other members. Posts are organized by subject into user-created boards called ”subreddits”, which cover a variety of topics like news, science, movies, video games, music, books, fitness, food, and image-sharing. As of July 2019, Reddit ranks as the 5th most visited website in the U.S. and 13th in the world”. [2]

2.1.4 Memedroid

Memedroid is also a website for user-generated content. It might be not as popular nowadays as it used to be in the past, however it still possesses its main feature, which is the ability to easily create your own memes and share them among others. Therefore Memedroid contains dozens of themes, formats which will be highly demanded while clustering, where variety of content improves the algorithm accuracy.

2.1.5 Imgur

[**Milestone 2**] ”Designed to be a gift to the online community of Reddit, it took off almost instantly, jumping from a thousand hits per day to a million total page views in the first five months.Imgur became widely recognized following its rise to popularity on social media websites such as Facebook, Reddit, and Digg. In October 2012, Imgur expanded its functionality to allow users to directly share images to Imgur instead of requiring images to gain enough attraction through other social media sites like Reddit to show up on the popular image gallery.” [3]

	Title	**Image**	**Url**	**Tags**	**Additional Data**
Reddit	✓	✓	✓	✗	upvotes, title, date
Imgur	✓	✓	✓	✗	upvotes, title, date, views
Memedroid	✓	✓	✓	✓	popularity, title, date
Twitter	✓	✓	✓	✓	text, retweet count, date
9gag			**Cancelled**		

Table 1: Raw sources content

[2] https://en.wikipedia.org/wiki/Reddit
[3] https://en.wikipedia.org/wiki/Imgur

2.2 Data collecting

[Milestone 1]

Besides memes themselves: tags, titles, urls and the popularity indicators will be extracted. Data will be extracted using Python libraries dedicated to scrapping (requests, BeautifulSoup) with the help of Reddit's and Twitter's API. The data, hoarded by webcrawlers and API's, will be in the form of JSON. Each sample will contain a few keys containing data in size varying up to a few kB.

[Milestone 3]

Collecting memes is fully automated via Apache NiFi. Each source is scrapped every 60 minutes. In table 2 we can see that single flow allows us to collect almost 250 memes.

Reddit	Twitter	Memedroid	Imgur	Total
187	32	10	10	239

Table 2: Average number of memes collected during one NiFi flow

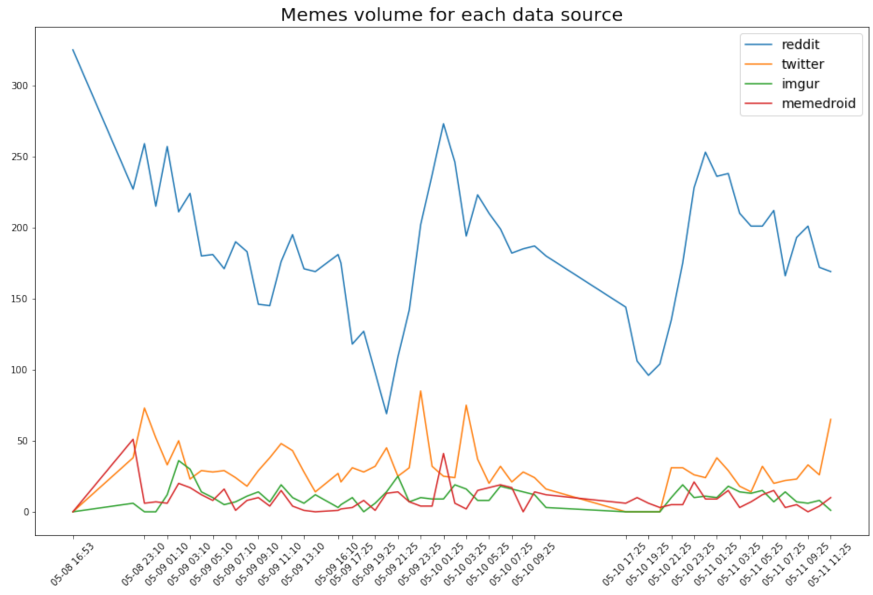

Figure 1: Amount of memes collected during each NiFi flow

As we can see in the figure 1, the reason for that is the fact that other APIs and collectors have their limitations in scraping volume. What is more, one can notice that our NiFi instance crashed on 10th of May. No memes have been collected during the over ten-hour time period.

Figure 2 shows that Reddit stands for more than 75% of total data we have in our storage. Both Memedroid and Imgur make about 5% of total volume each.

We have managed to collect over 14 thousands of memes in less than 3 days, which can be seen in the figure 3. Assuming that Apache NiFi flow will run constantly, we expect collecting over 105 thousands of memes every week.

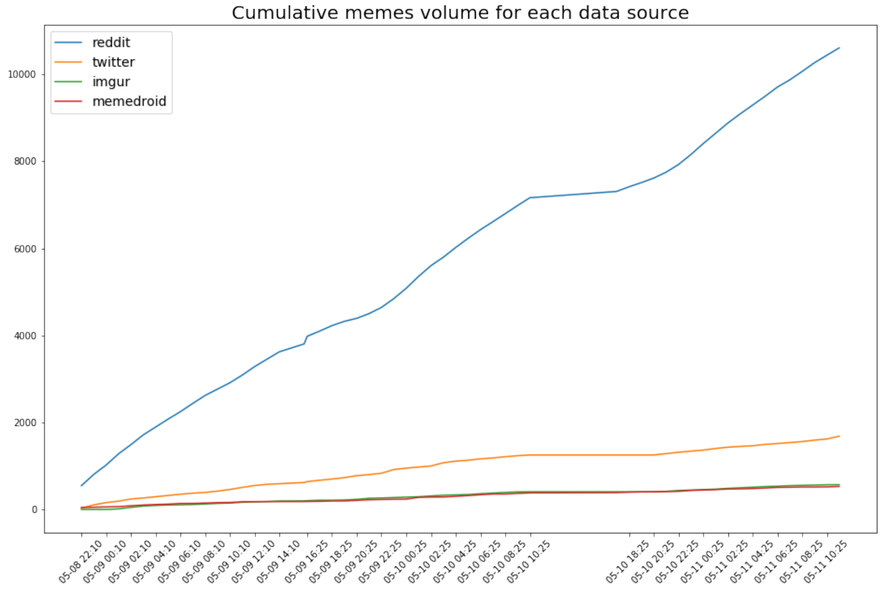

Figure 2: Cumulative amount of memes collected for each data source

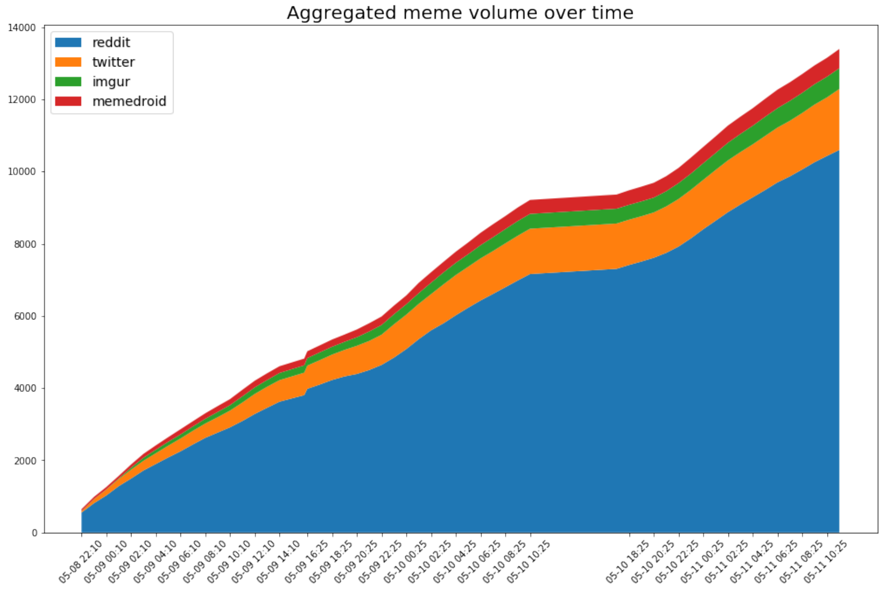

Figure 3: Total number of memes collected

[Milestone 4]

Over 80k memes have been collected since the beginning of May. This is significantly fewer /less than we expected. The reason for that is the fact that maintenance is very expensive.

As we can see in figure 4, dominant data source is Reddit. As mentioned above, all other scrapers have their limitations, while images from Reddit are being collected with no restraints.

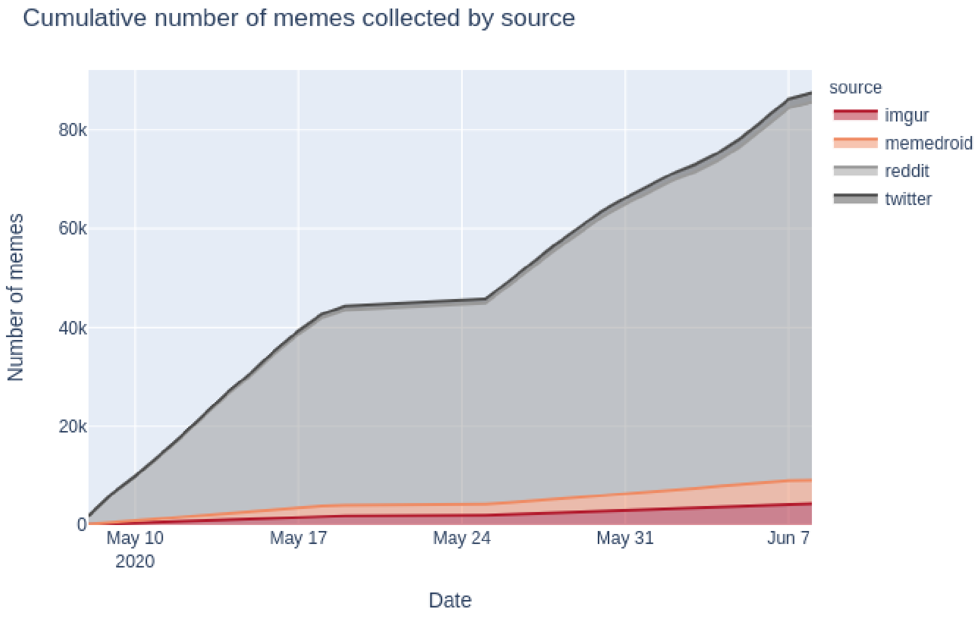

Figure 4: Total number of memes collected (website screenshot)

3 Architecture and tech stack

3.1 Architecture

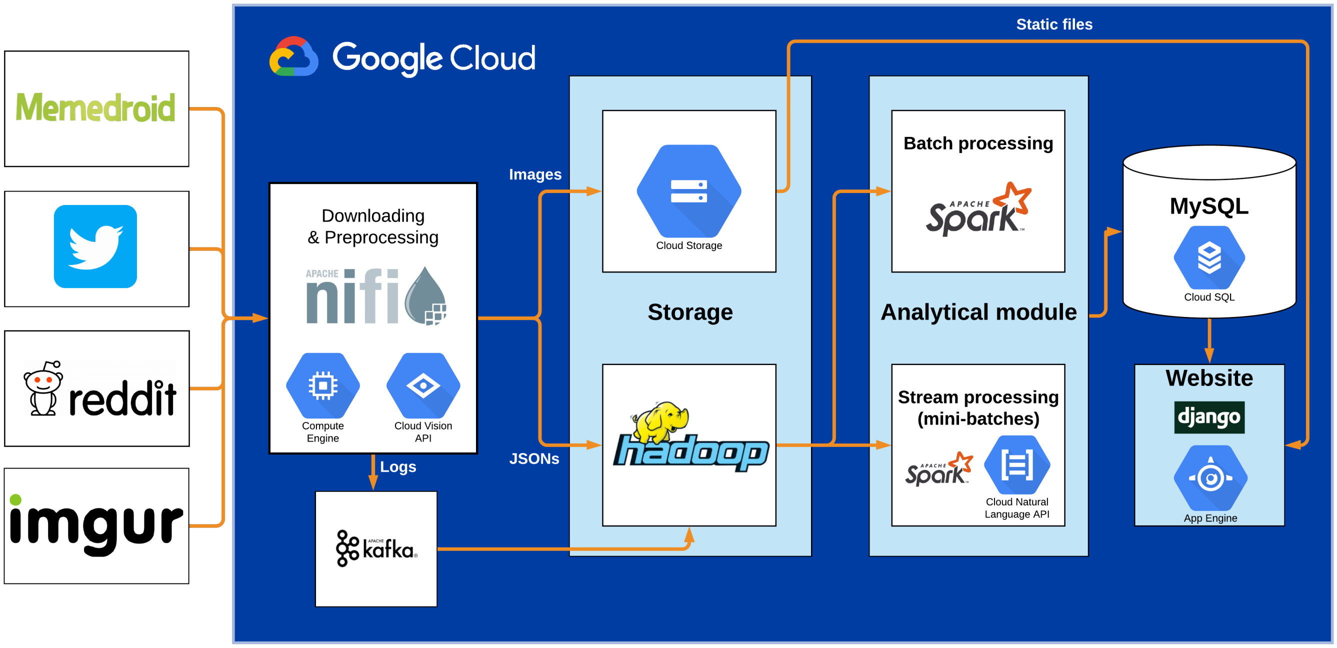

Figure 5: Model architecture

Project is running on Google Cloud Platform. Scraping and prepossessing is done using Apache Nifi, which is running on a GCP Compute Engine instance. Nifi turns on scrappers every hour, splits data into singular JSON FlowFiles, saves files on GCS bucket, runs Cloud Vision OCR and adds text to them. In the end Nifi merges JSON files into one. Another Processor Group is responsible for feeding Kafka brokers with Nifi logs splitted into three categories: INFO, WARN, ERROR.

Output of Nifi is split between two storages: Hadoop Cluster stores JSON files and logs, and Cloud Storage is storing images. On the previous milestone we considered using Google Dataflow for stream data, but since scrapping memes will work well with mini-batches we decided to only use Nifi.

Once a day, the clustering model is trained using Apache Spark library and meme statistics are updated. Once every hour, new memes are given cluster predictions by a pretrained clustering model.

All the results and links to pictures stored in the GCP bucket are kept in a relational MySQL database, connected to a website written in Django.

3.2 Tools

[Milestone 1]

- Google Cloud Platform
- Apache NiFi
- Google Dataflow
- Hadoop Cluster
- Spark
- Python
- Django
- SQL (MariaDB)
- Google Cloud Vision

[Milestone 3]

Added:

- Google App Engine
- Apache Kafka

Not used:

- Google Dataflow

Replaced:

- MariaDB with MySQL on Google Cloud SQL

4 Data processing

4.1 Preprocessing

All scraping processes have been uniformed, so that each meme has a list of common fields: meme id, image path, extension, source and url. On top of that, each meme has their source specific information. During NiFi flow, each meme gets an extra OCR text field.

JSON files are preprocessed using PySpark SparkSQL. For purposes of the descriptive statistics module we create one Spark Dataframe containing pair `meme_id` and `upvotes`. We also filter data, to remove obviously bad memes from the dataset. For purpose of clustering module we create two Spark Dataframes: one containing pair `meme_id` and `text`, and second containing pair `meme_id` and `tags`.

[Milestone 4]

We have started using `timestamp` attribute from `additional_data` field in scrapped json file to display correct time of creation of each meme on our website.

4.2 Feature extraction

[Milestone 1]

The main task is to extract text from each meme. To achieve such a result Google Vision will be used. Furthermore, data will be cleaned in order to avoid inappropriate content which might badly affect the later analysis. Moreover all duplicates will be removed.

The whole process might take such form:

1. extract each desired feature from JSON items,
2. download pictures in according to given urls,
3. remove pictures not considered as memes
4. extract texts from pictures
5. remove potential duplicates
6. forward data to analytical segment

4.3 Data Analysis

[Milestone 1]

The main task to be covered is to label each meme based on its content and format. To achieve such results OCRed content will be clustered and the most commonly used tags will be used as categories names. On the other hand Google Vision will perform clusterization of images to find the most popular formats for memes.

[Milestone 3]

Currently our Data Analysis consists of 3 main components:

- OCR
- memes clustering
- module of descriptive statistics

Optical Character Recognition

In order to extract texts from images we perform OCR algorithm. We use dedicated tool - Google Vision. Extraction is performed constantly (currently every hour), once Nifi confirms that image was uploaded to Google bucket. Such approach is cost-effective, ensuring that only stored pictures are taken into account (Google Vision API is for pay) and that all possible texts are extracted.

Memes clustering

[Milestone 3]

Memes are grouped in clusters according to texts they contain. Initially all extracted sentences are transformed into lists of words and preprocessed so that all common words such as 'I, or, and' are omitted. Such lists are passed through Word2Vec algorithm which transforms strings into numerical values. In order to prevent one value being more crucial than others, normalization is performed. Created features are passed to K-Means clustering algorithm which establishes clusters and assigns each observation to appropriate groups.

In order to assess the obtained model we use the Silhouette score which is a measure of cluster dispersion. Moreover we perform PCA as a way to visualize created clusters and monitor observations assignment.

Finally we look through memes' tags in order to assign a name (topic) to clusters. Following pictures and tables are presenting current model performance.

In the picture are: obtained clusters, hyperparameters (k - number of clusters, wn - number of numerical values after Word2Vec, Silhouette - value of silhouette score). Clusters are grouped by colors. Picture also contains numbers - which represent centroids of respective groups. It is worth mentioning that displayed observations do not fully represent real values, PCA explains more than 60% of variable variance.

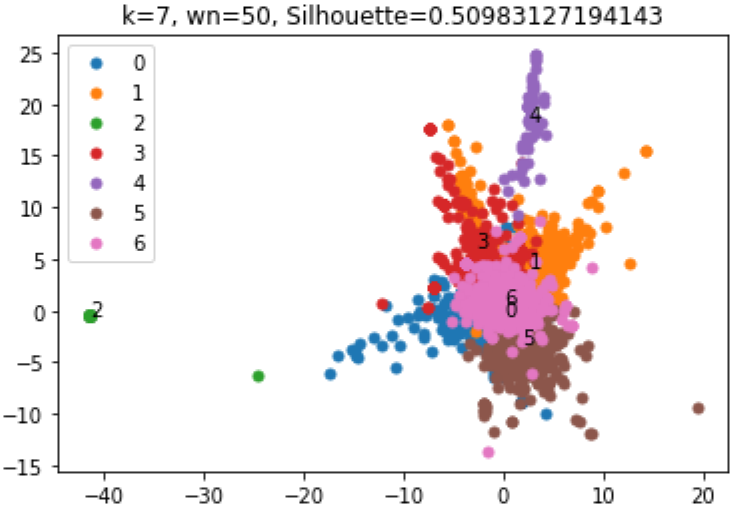

Figure 6: Clusterization with Word2Vec and k-means

First model used Word2Vec and KMeans. The dispersion factor was the best out of all obtained, therefore currently we will use this architecture to perform future clusterizations.

Moreover we have the most memes from Reddit. Its users have a "tendency" to share similar memes many times. These two factors play their role in model evaluation. For instance cluster number 4 is visibly distinct and is mostly associated with Scarlett Johansson's outfit on the red carpet. Perhaps we should associate such names to this cluster.

Afterwards, we went through memes' tags from each cluster to choose the most popular ones and assign them as cluster names. We obtained such results:

Cluster Number	Attached Name
0	funny
1	rpg
2	
3	
4	minimumrequirements
5	cyberpunk2077
6	minecraft

Table 3: Assigned cluster names

Unfortunately we have a very low number of tags (only memes have them, yet none from reddit where most of our data comes from), therefore we cannot expect our model to efficiently assign cluster names. We expect this problem to be solved when more data arrives.

[Milestone 4]

The only hyperparameter that was changed is the number of clusters. We increased it to 10 with to hope to more precisely distinguished different classes. We obtained such results:

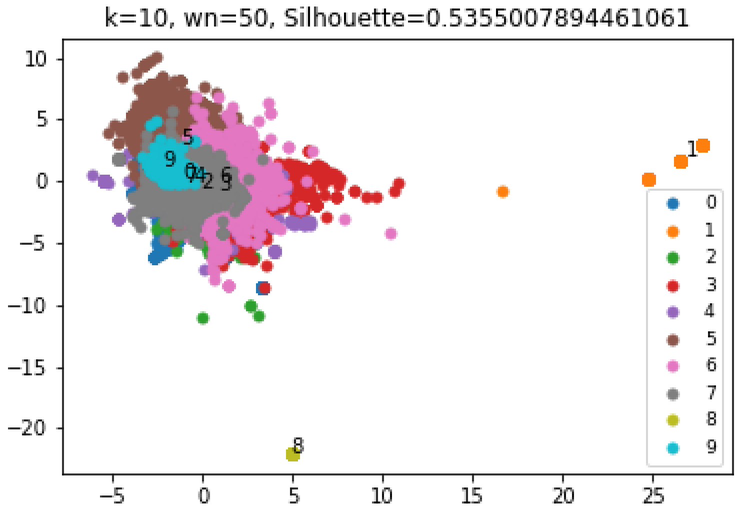

Figure 7: Clusterization with Word2Vec and k-means

Unfortunately, two first principal components explain around 33 % which indicates that the above plot do not represent the real data well. However, based on our observations the newer model is better at distinguishing separate classes. The silhouette indicator slightly improved. Next is the table with currently assigned clusters.

Cluster Id	Attached Name	Number of observations
0	other	24640
1	myart	1837
2	humor	34625
3	themoreyouknow	6527
4	coronavirus	7209
5	wholesome	9351
6	comments	5297
7	quarantine	18282
8	jjba5	391
9	hannigram	208

Table 4: Assigned cluster names

To summarize, despite countless trials we could not establish a model that would be precise in terms of clustering by themes. We mostly dedicated it to data being chaotic and unpredictive as well as countless number of tags that could be attached to every single meme. However we noticed that our model recognizes similar memes from different sources and subscribes them to the same cluster.

Module of descriptive statistics

[Milestone 3]

Memes from Reddit, Imgur and Memedroid are collected with upvotes data. Memes from Twitter have likes. Numbers from each site have different meanings, because of the varying number of users on each site. Therefore for each meme we want to analyze its performance compared to other memes from its source.

[**Milestone 4**] It is now possible to compare meme score against other memes from selected source. Example is shown on the picture below.

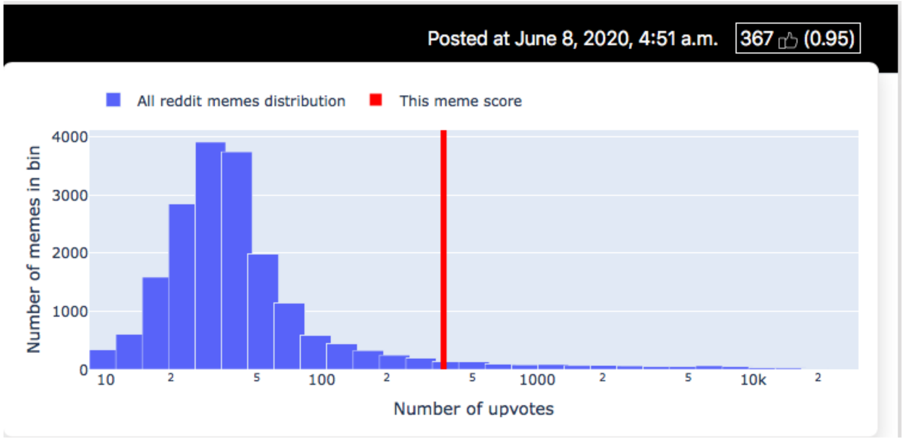

Figure 8: Example or comparing meme score against other memes.

We have also created a interactive Statistical Analysis Module with various plots displaying statistics about memes acquisition, sources, clusters etc.

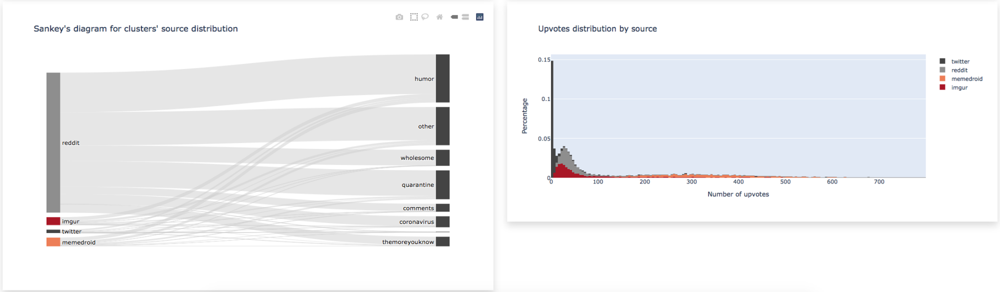

Figure 9: Two plots from Statistical Analysis Module.

5 Product

Our product is a web based application that allows users to browse memes (by default, newest memes will be displayed), view statistics about them and filter them by category, percentile score, source and date. Users are also able to view panels with information about collective statistics about meme sources, clusters and acquisition. Memelysis website is written in Django and it is working on Google App Engine, so it is highly scalable. We use Google CloudSQL MySQL database and Google Storage for website data storage.

[**Milestone 3**]

Our website currently displays memes and statistics about them. Other modules will be ready for Milestone 4.

[**Milestone 4**]

5.1 User interface

5.1.1 Memes browser

Main website is a stream of memes that user can browse. On the top bar there are sliders and boxes where user can select criterions by which memes should are filtered. The following options are included:

- Selecting minimal Virality Factor,
- Selecting mode of sorting (newest, oldest, best, worst),
- Selecting category,
- Selecting source,
- Combination of the above.

Besides that, one may enjoy the view into our data by exploring analytics subsection which is described in the following part.

5.1.2 Statistical Analysis Module

Analytics website delivers dashboard enabling users to monitor traffic on Memelysis. The following charts are included:

- Meme sources' distribution,
- Cluster distribution,
- Average amount of memes collected,
- Cumulative amount of memes,
- Upvote distribution,
- Diagram presenting source distribution for each meme.

6 Task distribution

Task		Michał	Jakub	Maciej	Krzysztof
Project of the solution		✓	✓	✓	✓
Implementation	Scrapers	✓	✓	✓	✓
	NiFi	✓			
	Hadoop cluster maintnance			✓	✓
	Data preprocessing				✓
	Data analytics			✓	
	Data visualisation		✓		
	Website	✓			
Tests	Scrapers unit tests	✓	✓	✓	✓
	Website tests	✓			
Documentation		✓	✓	✓	✓

Project of the solution was exhaustively discussed by all the members of our team. Implementation of NiFi module was done by Michał, backend Hadoop preprocessing and analytics was done by Maciej and Krzysztof and frontend website and visualisation was done by Michał and Jakub.

7 Testing

7.1 Unit testing

Our scrapers were unit tested for the second milestone (raport is included on the github repository in folder 2_Milestone).

7.2 Performance testing

We have performed performance testing of our website using Apache JMeter (test scheme is included in the Github repository). We have tested, that when 30 users are trying to use our site simultaneously, it is still working correctly and gives 0% of bad responses. This way we have checked that autoscaling of our website is working correctly.

7.3 Functional testing

We have conducted 7 tests of such functionalities like:

1. page loading time;
2. images display correctness;
3. category selection result;
4. source selection result;
5. minimal virality factor selection;
6. sorting options;
7. distribution plot display correctness.

Our assumptions were to provide access to memes dashboard in time less than 5 seconds. Images have to display correctly, selection of category, source or minimal virality factor (VF) results in adequate filtering of memes. After choosing sorting option user is able to see only memes with VF over the treshold. After click on likes account user should see a popup window with distribution plot of memes' upvotes from the same source.

8 Conclusion

[Milestone 3]

Task that we want to complete for finale milestone:

- Adding panel with collective statistics about meme sources.
- Enhance quality of predictions of clustering module.
- Displaying cluster name prediction on website.
- Adding filtering by date, source and percentile score to the website.

[Milestone 4]

All the above expectations are met in the last milestone.

We delivered expected product. Users might enjoy our website and search for their favourite content. Moreover, with the use of functionality test we verified that the user experience is positive.

Within the project we used tools dedicated for BigData solutions such as NiFi or Spark. We implemented our application within a cloud to maintain high availability and performance of our product.