**Facebook Scraping**

**Restricions**

Facebook scraping was the major goal. Unfortunately we run into problems because of the way the FB website is designed. Facebook has been developed with a very protective approach. Thus, scraping data is a very hard procedure.

When it comes to scraping profiles, there are some apps out there that can do the job of mining basic information that is being given freely by the people. Such applications are Octoparse, Phantomburst and others. Same thing happens with Facebook groups where you can mine public posts and comments by users.

Unfortunately, this doesn’t happen with Facebook Pages. For unknown reasons, Facebook pages’ data are not being served freely by Facebook’s API and hence, no free or paid web software gave us the opportunity to mine such data, nor were we able to serve ourselves from their API. The only think you can have from Pages API is posts data from pages you actually own or control. Being the moderator of the biggest Pharma FB pages was beyond the scope of this dissertation!

Under such circumstances, there was no other way but to create a scraper. A nice tool is the webscraper chrome plugin which has a very easy to use interface that let’s you mine data and make automations. Another tool that we used for scraping was octoparse.

Octoparse has a windows based interface where you select elements and octoparse automatically scrapes all the elements like the ones you selected.This way, you can easily and without any coding knowledge automate a procedure and scrape almost any website you want. But Facebook has some irregularities. First of all, Facebook has random class names for its posts. This way you cannot dictate any class name to be scraped because no element has the same name in the html structure. Octoparse has some AI mechanisms and it managed to understand the similarities but up to a point. We still had a major loss of scraped elements.

Webscraper Chrome plugin was much better at this job, finding all elements after an automatic scrape to the bottom. But then, then problem was that we couldn’t have the like, comment and share count aligned with each element. The results were terrible and it was impossible to continue with this tool.

Another alternative would be to scrape data by ourselves using Python modules but we wouldn’t be able to bypass the 2FA and confirmation procedures of FB with a testing browser (selenium).

The last resort was to device some vanilla Javascript directives in order to scrape that data. We managed to get the correct information for each post but then, the problem was that as the chrome browser was automatically scrolling, the Facebook was creating the new posts on the fly and we couldnt reference them. The result was dissapointing and the only way we could solve was maybe trying an sync auto scrolling function with delays but due to the lack of time we had to abandon the FB data mining.

**Linkedin Scraping**

Linkedin was the number one choice for statistical analysis. Even though we are using data extracted through images with the help of the Google Vision API and Instagram is the king of images, we prefered Linkedin because it’s a much more demanding social network for the companies. Furthermore, there is virtually no pharma company out there without a Linkedin Account.

After all instagram accounts were not that frequent and twitter has the text’s word count limitation and pictures were not that aboundant. Therefore we targeted Linkedin for combining the best of both and shows the most profesisonal character, while we found easy ways to mine the data. After all, we knew that linkedin accounts served as “default” social media accounts and any content posted there had a big chance of being shared in other platforms as well.

**Tools and procedure**

The initial data mining procedure was pretty simple and straightforward. No coding was needed at all as Phantombuster.com was chosen to do the work for us. The URLs of the linkedin accounts of 24 companies were fed as a CSV file into an already made phantombuster extractor (named “LinkedIn Activities Extractor”). The companies used were:

**Companies mined**

|  |
| --- |
| abbott |
| amgen |
| astellaspharmainc |
| astrazeneca |
| basf |
| baxter healthcare |
| bayer |
| beiersdorf |
| boehringer ingelheim |
| bristol myers squibb |
| chiesi group |
| eli lilly |
| gilead sciences |
| glaxosmithkline |
| janssen pharmaceutical companies of johnson and johnson |
| leo pharma |
| merck |
| merck group |
| novartis |
| novo nordisk |
| pfizer |
| roche |
| sanofi |
| ucb pharma |

As a result, phantombuster returned a csv file, containing 6924 rows, i.e. one for each post of all the companies along with the following data:

|  |  |
| --- | --- |
| profileUrl | This is the last part of the full linkedin account url, practically it is the name of the company |
| error | this is a field returning any errors, but actually returned none |
| timestamp | This is the time and date that mining happened. Example value: 2022-02-05T15:39:14.366Z |
| postUrl | This is the post's URL around which most of the processing happened until we came up with the final SPSS data |
| action | This field returned only the "Post" value and thus, was never used |
| imgUrl | The URL of the image of the post. Article, text, video and poll types, returned blank cells |
| type | The type of the post. Possible values: Article, image, poll, text, live video, video (external source), video (internal source) |
| postContent | The filed containing the string of the text of the post. |
| likeCount | The count of likes each post received |
| commentCount | The count of comments each post received |
| postDate | The date of the post. Unfortunately linkedin returned d/m/y before present day so no seasonality could be taken into account |
| viewCount | This field returns the number of views a post received. Since no account was belonging to us, there were no rights and we got only blanks |

These data acted as a base for the rest of the mining regarding image and text properties.

**Readability** Next step was to calculate the readability for each post’s content. We started by testing free or freemium apis on the RapidAPI.com platform in order to find something reliable. Due to the cost structure of these APIs, we decide to call it a day and choose a SAAS solution with an easy to use interface.  
 For this, we used another web service, called <https://readable.com>/. Readable offers a wide range of readability indices but we chose to use SMOG (because USDA advices the use of it in the drugs’ leaflets) and Flesch Kincaid Grade Level because it’s the most popular and SMOG is not very accurate with small texts.

Again, a CSV with the content of each post was fed (wherever text content existed) to the platform and a final CSV with all the indices was returned from which only SMOG and Flesch were used as explained above.

We decided to use both SMOG and Flesch-Kincaid Grade Level in our thesis because SMOG, eventhough officially suitable for healthcare material, it is supposed to be working better in texts with more than 30 sentences Something that it’s virtually impossible to come across in a LinkedIn post.

We ended up with 2 different numerical variables of scale type. This would be hard if we wanted to have some descriptive statistics so we decide to further classify both indices according to an “easy-medium-hard” rating. More specifically, for SMOG this formula was used in our excel file: =IF(G3<=9,"EASY",IF(G3<=14,"MEDIUM","HARD")), while for FKGL, =IF(I3<=6,"EASY",IF(I3<=12,"MEDIUM","HARD")) giving us the following table:

|  |  |  |  |
| --- | --- | --- | --- |
|  | EASY | MEDIUM | HARD |
| SMOG | -9 | 10-14 | 15+ |
| FKGL | -6 | 7-12 | 13+ |

**Images Processing**

We wanted to use a stable and reliable tool for images in order to extract elements of them plus the color ID of the pixels. There was no previous experience about it so we started searching in Google. We ended up in RapidAPI.com again where we found quite a few APIs that were only analyzing the similarity between two or more images or giving data about the image like the coding of the image etc. Some of these APIs were: Face Similarity, Similarity, CLIP Text-Image Image-Text Similarity. They were all instable and unreliable with results that was easy to prove wrong simply with a glimpse of a naked eye.

**Vision API**

So we ended up with a Google solution with which we could process the images for colors, faces and emotions. The service for all of these is called “Vision API”. We used python to extract these information.

More specifically, one python script (python version: 3.10) was devised in order to extract the faces number (if any) in all of the pictures. The maximum number of detected faces is 10.

Another one was written so we could fetch the feelings in the faces of each picture. Vision API returned a possibility for each of four feelings inside each photo for every face there. The 4 feelings are: Joy, Angry, Sorrow and Surprice. These data were not well suited for SPSS so we dicided to manipulate in an excel file and the final SPSS data had a boolean field for the presence of each of the 4 feelings. This means that if even 1 face in a picture had a possibility or certainty of showing Joy, then Joy field was True. Same for the rest of the feelings.

A person holding a camera

Description automatically generated

We can see an example of what it was returned after our own process. Here you see the number “1”, telling us that Google Vision Api found 1 face whithout prominent emotions.

A person smiling and holding up her hand

Description automatically generated with low confidence

Here is another example where we can see the presence of Joy (very\_likely) in the face of one person.

After the feelings and the number of faces, we started mining color information. Vision Api has another endpoint that returns the 5 dominant colors of each picture. Again, as with faces numbers and feelings, a TXT file with the image URLs was parsed by a python script and then was fed into the Vision Api for processing. Again, because of SPSS limitations, we kept only the most dominant color in hex format. Of course the result was thousands of unique HEX colorcodes that could not offer anything in any statistical analysis. This is where we realised that we had to classify the colors.

**Restrictions with images**

We faced a major restriction when it came to images. We realised that we could only access the posts that had an image that belonged to the linkedin domain, i.e. posts with images, or the first frame of a gif. If it was an image of an external URL, phantombuster returned a “not accessible” blank page when opening that image URL. Out of 6523 images, a total of 3525 were had to be classified as “Not Accessible” This restriction influenced the following metrics:

|  |
| --- |
| faces no |
| face exist |
| SORROW |
| SURPRICE |
| JOY |
| ANGER |
| Text Existance in Photo |
| Image Text Word Count |
| Brand Name in Photo Text |
| Dominant Color Names in Image |
| Html4\_color |

Of course no final result got harmed cause there was case selection in any statistical analysis that involved any of these variables.

**Color Classification**

As discussed previously, we used “webcolors” to classify the dominant colors extracted from the images by using the Vision API of Google which gave us thousands of different hex color codes. In the final SPSS we used the HTML4 protocol in order to classify the colors and get 16 different values. But, outside our statistical analysis, we also used the CSS3 protocol in order to create some visual representation aids to understand the colors used in a better way. We decided to create these treemaps by using the Plotly module for Python (https://plotly.com/python/plotly-express/) .

The creation of an AI machine learning algorithm from scratch was way beyond the scope of this project so we used a pre-made Python module (webcolors - <https://pypi.org/project/webcolors>/) that did the job in a great way. This module had a funciton to classify according to both css3 or HTML4 classification of colors. The css3 classification (<https://drafts.csswg.org/css-color-3>/) was used to create visual treemaps for each company and collectively. This way, we can take a look at the most frequent colors that were used in the images. The downsize of this, was that css3 has more than 100 colors and that way we wouldnt be able to make a statistical analysis inside the SPSS. This is why we used the HTML4 coding that has less than 20 and this one was used for our statictial analysis procedure.

Eventually, we ended up with only 16 unique colors for html4 and 124 for css3.

**Final SPSS-ready xlsx file**

The software used for the statistical analysis was IBM’s SPSS 27 with the basic package.

To summarize the full procedure:

1. Mining the linkedin posts with phantombuster and exporting the reults in xlsx
2. Feeding the mined posts’ text contents in the exact order into the readable api.
3. Extracting the readability results and pasting them in the exact order as new data in the original xlsx file
4. Feeding the mined posts’ images URLs in the exact order into a txt file in order for the Python scripts to parse them and process them with the Google Vision API
5. Pasting the results of each of the three API scripts into the original file as extra data
6. Copying the dominant color data (bulk number of HEX codes) from the original file into a txt file in order to be parsed by the 2 python scripts that were classifying the colors. The results were pasted back into the original excel in the exact same order.
7. Classifying the SMOG and FKGL ratings to the “Easy-Medium-Hard” ordinal variable.
8. Counting the hashtags inside each text.

Now, we had a full xlsx file with all the data needed. The following are the column names of it:

|  |
| --- |
| Company Name |
| postUrl |
| Type |
| word count |
| likeCount |
| commentCount |
| SMOG |
| SMOG RANK |
| Flesch Kincaid Grade Level |
| Flesch Kincaid RANK |
| Agreement between readability ratings |
| faces no |
| face exist |
| SORROW |
| SURPRICE |
| JOY |
| ANGER |
| Text Existance in Photo |
| Image Text Word Count |
| Brand Name in Photo Text |
| Dominant Color Names in Image |
| html4\_color |
| Number of Hashtags in Post Text |

We now had to transform this data to a more SPSS friendly type. So we created another xlsx file that we named coded\_for\_spss.xlsx and had the following columns:

|  |  |
| --- | --- |
| Company Name | Name of the Company (string) |
| company | Numeric Value (e.g. 1 = abbott, 2 = amgen etc) |
| postUrl | The URL of the post |
| Type | The type of the post. Possible values: Article, image, poll, text, live video, video (external source), video (internal source). Linkedin source video types are the posts that have an uploaded video. External source video types are the post that have a link to a video platform (usually YT) |
| post\_type | Numeric Values (e.g. 1 = artile, 2 = image etc) |
| word count | Number of words inside text (0 included) |
| likeCount | Number of likes a post got (0 included) |
| commentCount | Number of comments a post got (0 included) |
| SMOG | SMOG rating. Outlier of “3.13” value was found to be the default value of the posts that had not a single word (wordcount = 0). We selected case with more than 0 words for any descriptive or statistical analysis involving SMOG |
| SMOG RANK | Easy/Medium/Hard values |
| smog\_rank\_spss | Numeric value of ordinal type (=IF(J356="EASY",1,IF(J356="MEDIUM",2,3)) |
| Flesch Kincaid Grade Level | FKGL rating. Outlier of “-15.59” value was found to be the default value of the posts that had not a single word (wordcount = 0). We selected case with more than 0 words for any descriptive or statistical analysis involving FKGL |
| Flesch Kincaid RANK | Easy/Medium/Hard values |
| flesch\_rank\_spss | Numeric value of ordinal type (=IF(M356="EASY",1,IF(M356="MEDIUM",2,3))) |
| Agreement between readability ratings | AGREE/NOT values based on weather the two indices are both on the same level (easy/medium/hard) or not |
| agreement\_spss | Numeric Value of Nominal Type (=IF(O356="AGREE",1,0)) |
| faces no | Number of faces in an image. Blanks for not accessible ones and “no image” included. Case selection used in SPSS wherever needed |
| face exist | String Value. YES/NO/No Image/NA |
| face\_existance\_in\_photo | Numeric value of Nominal Type (=IF(R2="YES",1,IF(R2="NO",0,8))) 8 was used for the no image and not accessible ones and got treated as a missing value in SPSS. |
| SORROW | YES/NO |
| SURPRICE | YES/NO |
| JOY | YES/NO |
| ANGER | YES/NO |
| sorrow\_spss | Numeric Value of Nominal Type (=IF(T2="Yes",1,0)) |
| surprise\_spss | Numeric Value of Nominal Type (=IF(T2="Yes",1,0)) |
| joy\_spss | Numeric Value of Nominal Type (=IF(T2="Yes",1,0)) |
| anger\_spss | Numeric Value of Nominal Type (=IF(T2="Yes",1,0)) |
| Text Existance in Photo | YES/NO/No Image/Not Accessible |
| text\_existance\_in\_photo\_spss | Numeric Value of Nominal Type (=IF(AB2="YES",1,IF(AB2="NO",0,8))) |
| Image Text Word Count | Word count in number format of the text found in each image (No Image/No text/Not Accessible included). Case selection was used in SPSS wherever neede. |
| Brand Name in Photo Text | TRUE/FALSE/No Image (not accessible ones were classified to FALSE, so we were selecting cases in SPSS based on other variables and had a valid result) |
| brand\_name\_photo\_text\_spss | Numeric Value of Nominal Type (=IF(AE6721="TRUE",1,0)). Again, case selection based on other variables helped with results that were “0” but yet belonged to not accessible or “no image” posts |
| Dominant Color Names in Image | Css3 color classification (blanks for no image, not accessible and wrong color code included). This is why we get less values than the sum of image post type posts. Because some images were either not accessible or returned a wrong hex color code when parsing. |
| html4\_color | html4 color classification (blanks for no image, not accessible and wrong color code included). This is why we get less values than the sum of image post type posts. Because some images were either not accessible or returned a wrong hex color code when parsing. |
| html4\_color\_spss | Numeric Value of Nominal Type (e.g. 1 = aqua, 2 = black etc) blanks signify the wrong color codes, not accessible and no image |
| Number of Hashtags in Post Text | Simple count of hashtags inside the post |

<https://drafts.csswg.org/css-color-3>/

~~Πως καταλήξαμε στα απ και αν μπορώ να βρω ποια άλλα είχα χρησιμοπιοήσει~~

~~Μεταβλητές στο σπσσ για τα χρώματα και το ημτλ4~~

~~Να γράψουμε τους κωδικούς του εξέλ για τις μεταβλήτες~~

~~Για τον σμογκ και τον φλες πως τα χωρίσαμε σε εύκολο μέτριο δύσκολο~~

~~Πώς καταλήξαμε σε αυτές τις εταιρίες~~

~~Να αναφέρω τον πίνακα με τα σόσιαλ των εταιριών~~

~~Να προσθέσω έκδοση σπσσ και παύθον~~

~~Πάηθον να προσθέσω το μόντουλ που χρησιμοποίησα~~

**Companies Chosen**

We decided to choose some famous and financially stable multinational corporations of the Pharmaceutical industry. There was not a certain criteria that had to be met for a company to enter the list. It was more of a subjective opinion regarding the fame and success of the company.

The initial scope included the Greek branches of these companies (wherever existed) plus companies that are Greek but, during the research, we realised that Greek companies or branches had much fewer social media than the ones of mother companies and there was no consistency between these companies. Eventually, we ended up with the following companies that gave us 6924 linkedin posts that were finally analyzed

|  |
| --- |
| abbott |
| amgen |
| astellaspharmainc |
| astrazeneca |
| basf |
| baxter healthcare |
| bayer |
| beiersdorf |
| boehringer ingelheim |
| bristol myers squibb |
| chiesi group |
| eli lilly |
| gilead sciences |
| glaxosmithkline |
| janssen pharmaceutical companies of johnson and johnson |
| leo pharma |
| merck |
| merck group |
| novartis |
| novo nordisk |
| pfizer |
| roche |
| sanofi |
| ucb pharma |

**Social Media Channels**

As one can see in the crosstable below, all of the companies have a linkedin account. Instagram is the least common social media channel while Twitter is the second most frequent. Youtube is also there for almost 90% of the companies analyzed and we were ready to use the Google Video API for analyzing the videos of the companies but, both time restrictions and the scope of this dissertation made us abandon the YT channel. Facebook restrictions and Twitter will be discussed later.

Table

Description automatically generated