NEWS SEARCHING

Matteo Silvestri
ID: 1774987
Master in Engineering in Computer Science
silvestri.1774987@studenti.uniroma1.it

September 16, 2020

1 Introduction

After searching some datasets sources in the internet, i found out the public articles dataset of http://mashable.com, this is one of the most important news website in USA. In this project i study the best formula to have a successful interactive article (having multimedia content) and how the day of the pubblication or the field(Busiuness...ecc) it affect on the shares.

2 Dataset Info

I used the online news popularity data set of http://mashable.com available from UCI Machine Learning Repository at https://archive.ics.uci.edu/ ml/datasets/Online+News+Popularity. Mashable news dataset consists 61 heterogeneous features about the associated statistics of 40000 original news articles released by Mashable during a two years period from January 2013 to January 2015. Each element has an ID that corresponds to the URL of the news. The most interesting attributes that i selected for our visuals are: Number of words in the article, Number of external links in the article, Number of images in the article, Number of videos in the article, Article argument, Publication day, Number of shares. First of all i worked on the dataset to make it more comfortable to work with, for example there were 7 attributes for the day (isMonday, isTuesday ..) and i made a single column were the number from 1 to 7 corresponds to the respective day of the week, and the same for the article argument, in order to have a more simple dataset comprension.

2.1 Preprocessing

In order to make the dataset more readable and to have more significant views for our scope, i cut the initial database in the following way:

- Cut off the data with missing parameters
- Grouped all the isDay attributes in a single attribute called Day, which can have values from 1 to 7.
- Grouped all the isArgument attributes in a single attribute called Argument, which can have values from 1 to 6.
- In order to have a better visual representation of the success of an article, i divided the data in two categories: Popular(Orange) and Unpopular(Green). Each article with the number of shares greater than the average is considered successful.

Therefore i obtained a version of the original dataset which counts 7998 tuples with 11 attributes for a total of 87978 values to manage(After the preprocessing)

3 Launch It

I used a python framework(Flask) in order to run locally the project. Once the server is started, by launching from the terminal server.py through python, the visualization can be seen launching a browser (preferably Firefox) and visiting http://localhost:5000.

4 Main Visualization

I built 4 different kinds of graphs, described below, i try to show to the user many different perspectives of the data, in order to make simple the searching of an article inside through the dataset. In order to develop this project i used pandas, numpy and the D3.js framework, the interface work even if i try to resize the format of the screen, and i also add a form for add a new article, a text area where we will see the mean number of images selected from the user(only if i choose the mean scatterplot) and another one in order to search an article through the URL and highlight all the data about it on the others graphs.

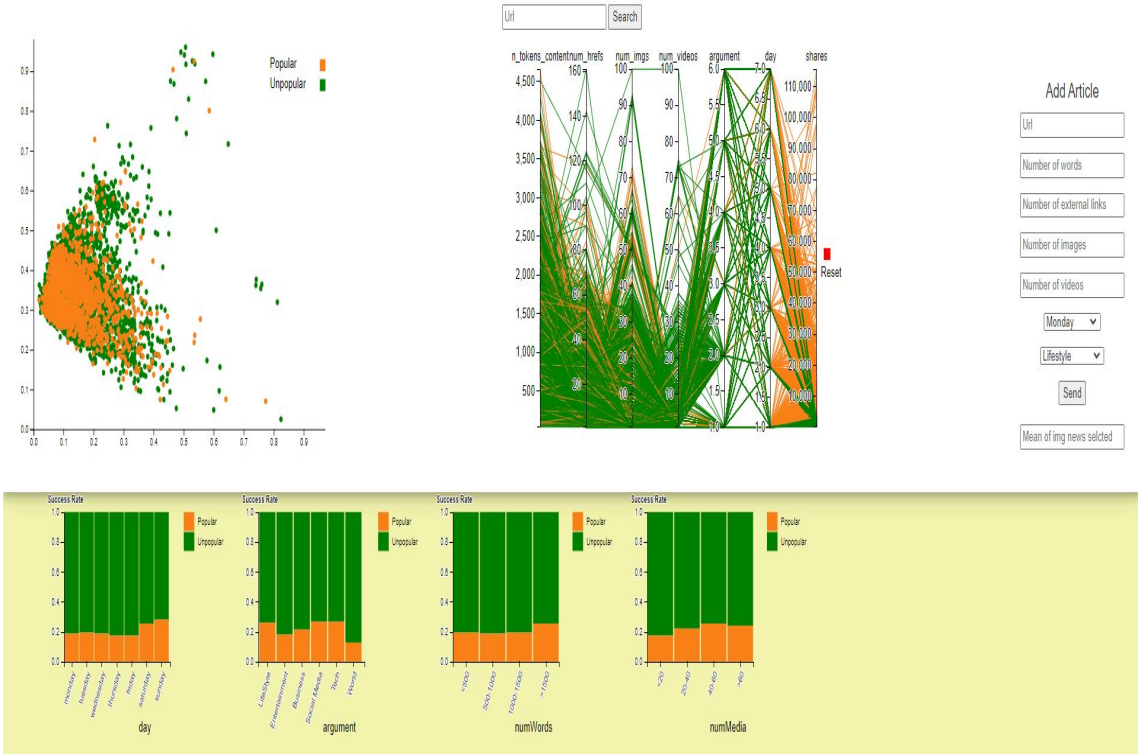

Figure 1: Main Visualization

4.1 Scatterplot

We have 2 different kind of scatterplot, that the user could choose from the suitable button, in general both the graph graphically represent the relation between points and to spot possible clusters, each point represent an article and it' s color means if the news is popular(orange) or unpopular(green), i will consider an article popular if has a number of shares higher than the average, viceversa unpopular. I applied a dimensional reduction algorithm(PCA). PCA don' t require a deep knowledge of the dataset and it is fast. I used **pandas** to read and parse the csv original dataset and **sklearn** to compute PCA values, saved in pca.csv. This procedure is done by the python function **pca.action()**. The user can mouseover on the points to see a preview of the link of the article, and with a click the point will become red and bigger and the artcile will be highlighted both in the stackbarcharts and on the parallel. In top right i put the legend that explain the color management and in the same time, if clicked, will highlight the respective point inside the scatterplot.

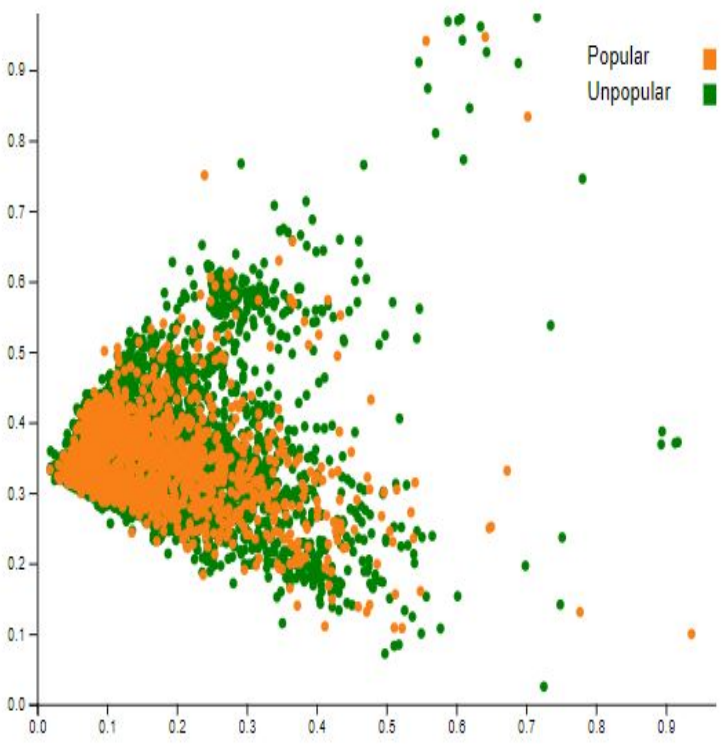

Figure 2: Scatterplot

4.2 Parallel

This is an efficient kind of view in order to represent data in a 2 dimension. In this graph we have 7 axes, each one correspond to a different parameter from the dataset:

- *Number of words*
- *Number of links*
- *Number of images*
- *Number of videos*
- *Argument*
- *Day*
- *Shares*

If an user wants can drag the axis for change the order, i add a brush function that allow the user to apply filter through the axes if he needs to focus on a specific range of some parameter. I add a reset button on the right side of the screen in order to reset the graph and deactivate the brush effect after an highlight from the others graphs.

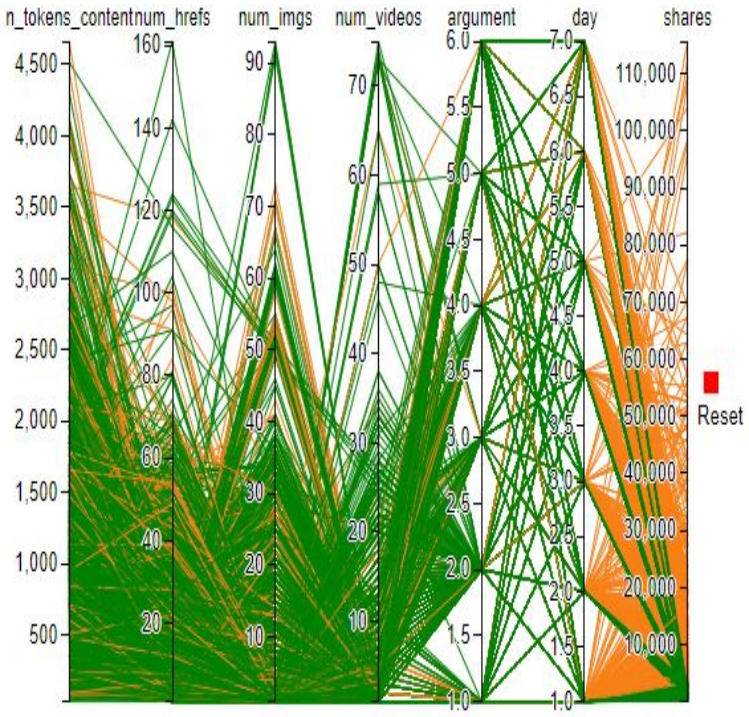

Figure 3: Parallel

4.3 Stacked Barcharts

In this kind of graph we have categorical attributes (days and argument) and the ranges of number of words and number of media (links+images+videos) of the dataset, having in the axis of the ordinates the success probability. This kind of visualization allows the user to see at the same time 4 different frequency distribution of both popular and unpopular articles. There is one column for each different value of the axis of the abscissas, composed by two sub-columns. The datas that i use are taken from **frequency.csv** and **instances.csv,** created at the launch of the server with two python scripts, that gives to us the number of interested instances and their frequency of success. I decided to use a tooltip that shows the user the value of a bar since datas are in percentage. In those stackbarcharts, bars can be filtered by clicking the legend in order to see popular or unpopular articles only. Each bar can be clicked, and it will be highlighted together with all points in the scatterplot that correspond to articles that have the same value for the attribute choosed, the corresponding classification of the clicked bar and all the corresponding lines in the parallel chart.

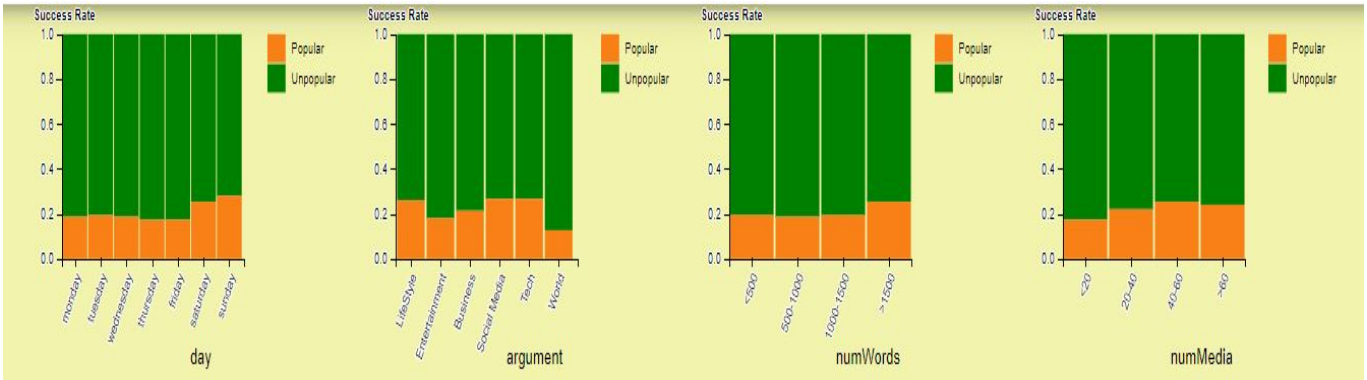

Figure 4: StackBarchart

5 Analytics

So for the analytics part i choose to add the form for add a new article adding some info(Url, Number of Words, Number of Links, Number of Images, Number of Videos, Publication day and Argument), i send to the server(through AJAX) the instances of the form where it computes a vector distance with the new article with the whole dataset, in this way i choose the article closer to the new one in order to study them and compute the shares of the new one. The new article will be inserted into the dataset and the server will sent to the user a response with the expected number of shares. Another analytics computed is the mean of the number of images of a cluster of articles choosed trough the scatterplot(the one that highlights more than one article) and print these values inside a textbox on the right of the screen.

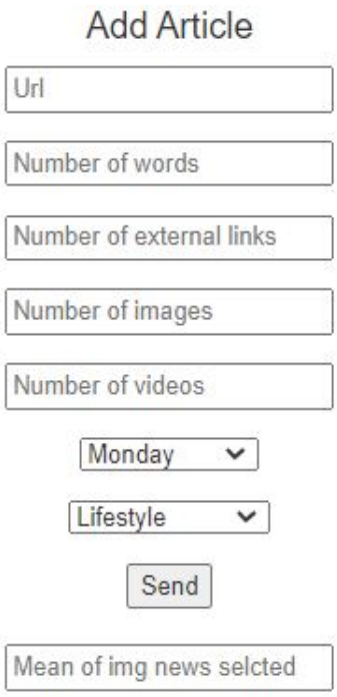

Figure 5: Form in order to add a new article

6 Conclusion

This project could be used in order to do a lot of different task from an user, like:

- Compute the mean of the images inside a cluster of articles
- Add a new article on the dataset and see where will be placed in all the graphs
- Highlights, from the stackbar, some specific articles with a specific range of words or a specific day of publication ecc....

Due to high number of tuples in some graphs the visualization could be a little bit crowded(if the number of tuples increase we have some crowding in the graphs), always for the high number of tuples inside the dataset we could find a minimal latency in the highlighting, but is still acceptable.

7 References

- Online News Popularity Data Set: **https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity**
- D3.js API: **https://github.com/d3/d3/blob/master/API.md**
- Scikit-learn: **http://scikit-learn.org/stable/documentation.html**
- Numpy: **https://docs.scipy.org/doc/**
- Pandas: **https://pandas.pydata.org/pandas-docs/stable/**