

**TASK**

**Exploratory Data Analysis on the Layoff**

**2020-2022**

**Data Set**

[](http://www.hyperiondev.com/portal/)

**Introduction**

Summary of the data set ----- LAYOFF ANALYSIS

The Project explores a Dataframe of Layoffs Data from 2020-2022, consequently to Covid-19 Pandemic and beginning of Inflation.

There are 11 columns and 1809 Rows

Original dataset can be tracked at <https://layoffs.fyi/>

#### **Columns Identification:**

Company = Name of the Company

Location = Location of the Company

Laid\_Off\_Count = Total count of Layoff

Percentage = Percentage of layoff

Date = Date of Layoff

Source = Data gathered from source

Funds\_Raised = Total Funds raised , according to the corresponding Stage

Stage = Financial Stage of the company (Fundings)

Date\_added = date the data were added into database

Country = Country where the company is located

List\_of\_Employees\_Laid\_Off = link to g\_docs of employee list

The analysis wants to explore globally the companies registering the highest layoff, which country was most affected and if there was a pattern on the redundancy process over the years 2020-2022.

Is there a correlation between the financial stage of the company and layoff?

Two Data cleaning method were used and compared as experiment for this study, the goal is to show whether the deletion of nan values could be complemented by a different approach if it should have been excluded a prior.

**DATA CLEANING**

It has been used **drop of duplicated rows** to make sure no duplicates rows would stay in the df. After checking the result, 1809 rows were returned, meaning that no duplicated data were found.

To visualise the relevance of Missing data in the dataset, a data-dense display from **Missingno** was applied: through the adoption of **Matrix, 3** columns seems mostly affected**.**

‘Laid\_Off\_Count’ & ‘Percentage’ columns seem missing mostly of their data, which raise the question of how to use the database and manage all these missing values.

The following columns are dropped:

* List\_of\_Employees\_Laid\_Off'
* 'Source'
* 'Date\_Added'

As they are not relevant for the exploration of this research.

MISSING DATA

To dive deeper into the first exploration level, the .isnull() function has been applied to find the total of nan values:

tot\_nan = df**.isnull().**sum()

The function reveal:

* **539** missing entries for ‘Laid\_Off\_Count’
* and **588** for ‘Percentage’
* ‘Funds\_Raised’ is missing **134**
* ‘Date’ missing **1**

**7.7 %** of the data in the dataset is found **missing.**

**This figure shows that most of the dataset is intact, however the amount of data unavailable affect considerably the most useful of all data: the laid\_off\_count is the reason we carry the research, and almost half of the column is gone.**

**Considering that 588 values are also missing from the Percentage, it means that if we remove all rows with nan values, we will definitely lose at least half of the dataset.**

**Two ways of handling data have been experimented and they will be presented**

**in this report.**

1. **One exploration will carry the data visualisation using half of the data set and dropping all the rows with nan values found in the columns laid\_off\_count and Percentage.**

**The reason on choosing this approach lay on the fact that the layoff is the main data to interpret, the rest could be considered marginal.**

1. **The second approach instead goes replacing the nan value with guessed values from the interpretation of the data we already have.**

**Choosing this approach save the marginal data like *the number of countries* and *locations* involved in the layoff and present in the data set. We also have more data about the company stage during the redundancy process.**

Firstly, was calculated the mean and the median of the Laid off column.

The **mean** and **median** are significantly different (mean = 194, median = 70), therefore **it will be chosen the median** to describe the centre of the distribution using the .fillna() function

df["Percentage"].fillna(0.18, inplace=True)

0.18 is the median values

Secondly was calculated calculated the mean and the median of the Percentage column.

The **mean** and **median** are in this case quite close to each other (mean = 0.27, median = 0.18) on a scale of Percentage from 0 to 1.

Our distribution has slightly bell shape but not quite, although the mean and the median are closer to each other, compared with the previous observation, they present a moderate distance.

Therefore, **it was chosen the median** to describe the centre of the distribution.

**The absence of the real layoff data limits manipulation, we must consider that when using laid\_off\_count as parameter of comparison, or Percentage, the first cleaning data approach is closer to the reality as explore 100% given data.**

**While the second data cleaning preserve better the exploration of the rest of the columns.**

The single missing **date** is not relevant as missing data, and it will not compromise the exploration of that column.

**Funds Raised** columns will be left as well, as it won't be used in the visualition.

EXPLORING OCCURENCIES OF DATA INCONSISTENCY

**One Error**: 'Date' columns is saved as object and needs to be converted to datetime64

**A new column was created : date\_parsed, with the parsed dates**

**df['date\_parsed'] = pd.to\_datetime(df['Date'], format='%Y-%m-%d')**

**df['date\_parsed'].dtype**

**Data inconsistency related to data entry.**

The numerical most relevant columns have been quickly checked to find if any data entry was inserted as string rather than numerical values.

**df['Percentage'].unique()**

**df['Laid\_Off\_Count'].unique()**

**df['date\_parsed'].unique()**

**After applying the unique Function, no error was found.**

DATA STORIES AND VISUALIZATIONS

**LAYOFF DISTRIBUTION**

#### **First difference between the two data cleaning analysis**

**The second char should be more attenable due to the fact that comes from 100% confirmed data. Noticing the x count: there are far more data in the imputation char, meaning that all guessed values really sum up and changed the prediction.**

**Chart, bar chart

Description automatically generated\_\_\_Chart, bar chart

Description automatically generated**

**graph 1 : Imputation graph 2 : Drop Nan**

#### Results

#### Most of the companies have moderate loss of workforce: under **50** people.

### LAYOFF DATA by COMPANY

First quick exploration of the most affected companies by Layoff.

It results that **Meta** registered the highest amount of employees let go during pandemic. The giant company is **followed by Amazon**, another big name.

In the top ten we read still some popular names:

Uber

Booking

Cisco

Peloton

Better.com

Carvana

Twitter

Bytedance

**1501 Companies** in total were found, and we immediately notice that a part for the top 2, registering an average of 10500 staff discharged, the amount of layoff significantly drop below 5000 with Booking.com.

**Comparing different Data cleaning results**

**Again The 2 chars show different result, small differences in this case.**

**The second char should be used for interpretation as Is based mainly on layoff count.**

**Chart, histogram

Description automatically generated\_\_Chart, histogram

Description automatically generated**

**graph 1 : Imputation graph 2 : Drop Nan**

We obtained similar result, and the data cleaning approach didn’t change the main results which see **Meta** and **Amazon** on the verge.

**FUNDS RAISED VS STAGE**

**Comparing different Data cleaning results**

Y axis = Funds\_Raised

**In this case, layoff count wasn’t used as main parameter of evaluation.**

**The first graph takes into account more data and this time are verified, as the Imputation method guessed layoff count and percentage but allowed us to maintain the marginal actual data, like Fund-raised in this case.**

Chart

Description automatically generated \_\_Chart

Description automatically generated

**graph 1 : Imputation graph 2 : Drop Nan**

Results:

**We still spot some minor differences in the result but for sure**

Series H

Series J

Are confirmed to be the Financial Stages with the highest cash available

Interestingly, **IPO** companies got a quite good amount of finance compared with other Stages.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### **Highest Layoff by Company's Stage**

**LAYOFF VS STAGE**

**Comparing different Data cleaning results**

Y axis = Laid\_off\_count

**The second char should be taken into more consideration to interpreter results.**

**We notice differences:**

**Ipo in the Imputation was markedly lower**

**Serie I considerably higher**

**Serie G and Seed inverted completely the results.**

Chart, box and whisker chart

Description automatically generated\_\_\_Chart, box and whisker chart

Description automatically generated

**graph 1: Imputation graph 2 : Drop Nan**

How the Financial stage affected redundancy? There is any correlation between the two of them ?

IPO

Serie J

Serie H

Were the most affected from layoff

We previously observed that Series G and H received the highest financial support and we calculated that Serie H followed with redundancy such numbers.

Serie G (moderate layoff) is quite distinctive for the registered count, with a similar result of Serie J (high layoff).

Further analysis is required to drag better conclusion as Highest stage it doesn’t seem to necessarily conclude with high layoff.

**AFFECTED INDUSTRIES**

**First graph to refer.**

**Spotted differences:**

**Chart, bar chart, histogram

Description automatically generated\_\_Chart, bar chart, histogram

Description automatically generated**

**graph 1 : Imputation graph 2 : Drop Nan**

**Finance** and **Retail** confirmed to be the most affected industries.

Not important differences with the variation of the further 3 Industries displayed in the 2 graphs, as somewhat **Healthcare, Food** and **Transportation** were similarly affected**.**

**LAYOFF DISTRIBUTION AROUND THE COUNTRIES**

**Layoff Countries distribution: Number of countries involved \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

The char counts the number of Countries registering layoff and return a distribution of layoff, presenting a panoramic of the world involvement.

**Total Layoff by Country.** The second char sum instead the total Layoff and therefore how impacted each country.

For the ‘Layoff countries distribution’, it has been selected the first char Pie (first on the left), assigned from the **Imputation cleaning method**, as it is counting only the countries registering Layoff. The corresponding char coming from **‘Drop Nan’ data cleaning** has lost some of the layoff data, since the Percentage nan rows were deleted, it might have happened that few Layoff were cancelled in the process too.

To analyse the ‘Total Layoff by country’ instead, the second char pie from ‘Drop nan’ cleaning method should be selected (second pie down on the right), as the guessed data were left out.

**Spotted differences:**

**Chart, pie chart

Description automatically generated \_\_ graph 1: Imputation**

**Chart, sunburst chart

Description automatically generated \_\_ graph 2: Drop Nan**

**I know this visualisation is not the best to identify the smallest portions, however for our analysis only the biggest slices of the pie are relevant.**

**Results:**

Most of the people in our data sample lost their jobs in **United States** (however, we know that we had far more data related to such Location) and **India**, which significantly registered the highest layoff

In **Uk** less companies adopted redundancy but more people lost their jobs compared with other countries

**UNITED STATES DATA**

To this point, we understood that the US data are the most relevant figures in our dataset to which it will be dedicated a brief more in-depth consideration.

**HIGHEST LAYOFF: TOP 10 LOCATION IN USUS \_\_\_**

**\_\_\_\_\_**

**Refer to the First graph. (Imputation method)**

**Spotted differences: \_\_**

**In the second analysis, Chicago figures on as 5th instead of Lost Angeles and Austin is out form the top 10. The main results are preserved however: top 4 is confirmed with both analysis**

**Chart, bar chart, histogram

Description automatically generated\_\_ graph 1: Imputation**

**MOST COMMON COMPANY STAGE**

**Chart, bar chart, histogram

Description automatically generated\_ graph 2: Drop Nan**

**Refer to the First graph. (Imputation method)**

To calculate the company stage it was used layoff count but the result was obtained from selecting only us companies from the dataset, then grouping them by Stage and ordering the values.

us\_stage = us\_companies.groupby(by='Stage').count()

us\_stage = us\_stage.sort\_values('Laid\_Off\_Count' , ascending=False)

**Spotted differences: \_\_**

**Serie C lost 1 position, found on the 3rd place instead of 2nd. SERIE B found in front of C when it was calculated behind in the Imputation method .**

AQUIRED lost 2 position to STAGE A and STAGE E.

**However, all these stage have very similar results and the overall order is not completely distorted.**

**Results:**

In **US** the **LOCATIONS** most affected result to be:

1. SF Bay Area

2. New York city

3. Seattle

4. Boston

5. Los Angeles

Companies in **IPO** financial stage were the most at risk, it is a crucial stage for a commonly and it shows how precarious are these companies in such stage, more than the **seeds** interestingly.

At the end of the Financial Risk classification, we notice **SERIE G** & **SERIE I** stage, To elaborate such position however we must consider that less company, not many, reach that financial level, that's why we see it at the end of the line. It doesn't necessarily that these company weren't affected by layoff.

Seed are found towards the end too, to consider however that still, these companies have less to lose and less resources and workforce to make redundant.

Interestingly we found **Serie C** more at risk than **Serie B** & **Serie D.**

**Series D** funding occurs when the business was not able to meet its targets with its **Series C,** and consequently it can mean that the business is now at a lower valuation. Being priced at a lower valuation is usually very negative for a business.

The same for **Series E.** This is, again, a very bad sign, and very few companies are going to survive to Series E funding.

Series E funding will only occur if the business still hasn’t been able to make up its own capital

but the business is still struggling to remain active and private.

Despite the definition, it seemed US **Series C** were in worse condition over the Pandemic.

**HIGHEST LAYOFF: TOP 10 LOCATION IN US**

**Refer to the First graph. (Imputation method)**

**Spotted differences : arrows indication**

**Chart, bar chart, histogram

Description automatically generated\_\_ graph 1: Imputation**

**Chart, histogram

Description automatically generated\_\_. graph 2: Drop Nan**

**Results :**

To re-dimension the **layoff**  and **Percentage** data, the graph on the left has to be paired with the yellow bar graph on the right, to understand the real loss of the company. In fact, despite Meta and Amazon shares big layoff number, overall, the workforce of the company wasn't so affected, as le than **20% of the labour** was cut off.

The **workforce** of these company is on average:

**Meta. 71 000 employees**

**Amazon. 1 million "**

**Cisco 72 000 "**

**Uber 22-29 000 "**

**Peloton 8 000 "**

**better.com 8 000 "**

**US INDUSTRIES LAYOFF**

**Chart, histogram

Description automatically generated**

It's relevant to observe that **1192 rows** were returned, meaning that 1192 Us companies were found in this database. During our analysis we should take into account this, as the data could be biased otherwise.

In **US** the **INDUSTRIES** most affected result to be:

1. Consumer

2. Retail

3 Infrastructure

4. Transportation

5. Real Estate

The sector **less at risk** were:

1. Aerospace

2. HR

3. Legal

4. Data

**WORLD DATA**

Last investigation related to world data would dive more into how much of the workforce an average company had to sacrifice, comparing previous data with any correlation found between Funds Raised and Percentage. The second part will analyse instead the trend of layoff, month by month happened over the years 2020-2022.

**Percentage Distribution**

Similar results despite the loss of Percentage data

**Chart, histogram

Description automatically generated\_\_** Chart, histogram

Description automatically generated

graph 1: Imputation Graph 2: Drop Nan

Most of the companies lost 20% of the **workforce**

**Median of Percentage-loss of Companies**

**Refer to graph 2 as Percentage comes from not the guessed data**

**Chart, line chart

Description automatically generated\_\_\_ graph 1 : Imputation**

**Spotted differences:**

Less companies under **20%** Percentage loss calculated on the second graph.

**Graph 2: Drop Nan\_\_ Chart, line chart

Description automatically generated**

On average, a company would expect to loose less than 0.3 **workforce (<30%)** of their companies consequently to the 2019 crisis

**Correlation between Funds Raised and Percentage of Layoff**

**A picture containing chart

Description automatically generated**

There is not a direct strong correlation between Financial Stage and Percentage-loss of a company, however, we don't have many samples of big companies.

Companies with the highest financial funds survived.

For some companies at the beginning of the financial stage bankrupt (loosing 100% workforce)

**TIME**

Breaking down the time frame available in the data set:

**OVERALL LAYOFF TREND**

**The general trend results very similar with both cleaning methods.**

**Chart, line chart

Description automatically generated\_\_Chart, line chart

Description automatically generated**

**graph 1: Imputation graph 2 : Drop Nan**

On a first sight it appears that there was a significant drop on 2021, the following chars will extract each year’s data.

**YEAR 2020**

**Text

Description automatically generated with medium confidence\_\_Chart

Description automatically generated**

**graph 1 : Imputation graph 2 : Drop Nan**

There is a significant drop after May 2020. The first spike happened after the first lockdown announcement.

**YEAR 2021**

**Chart, line chart

Description automatically generated\_\_Table

Description automatically generated**

**graph 1: Imputation graph 2 : Drop Nan**

No data available in 2021 using data cleaning method Drop Nan. Matching the previous graph, we could come the conclusion that maybe there weren’t a real drop of redundancy but instead a lack of data in our dataset for the 2021.

**YEAR 2022**

**Chart, line chart

Description automatically generated\_\_Chart, line chart

Description automatically generated**

Spike over the summer and before Christmas, surprisingly, time when usually companies tend to hire more staff. World expecting an economic crisis and the inflation started to take on.

**CONCLUSION**

**DATA CLEANING**

During the exploration of this Layoff-dataset, we compared 2 types of data, aligning 2 chars each time, being the result of a different **Data cleaning method. In the first instance** it was adopted Imputation for the Layoff\_count and Percentage columns. The second method instead made use of the Drop function to eliminate every rows not containing Layoff\_count and Percentage data. This approach costed to loose **7.7 %** of or dataset.

**LAYOFF TREND OVER TIME**

We found that in this dataset the data related to the time frame 2021 are missing. Overall we distinguish a significant spike in layoff at the beginning of 2020, corresponding with the first lockdown announcement. The situation stabilised after May 2020. Probably it took an uphill turn during the year 2021 but unfortunately we have no record of it.

During 2022 we didn’t assist to similar figures but another layoff trend took off during the 2022 summer and before the Christmas festivity, headlighting the theory that the Inflation and an economic crisis is starting.

**OTHER**

**Average layoffs count a loss of 50 employee. The range concentrate around 100-50.**

**Amazon Meta Cisco highest layoff d**

**THIS REPORT WAS WRITTEN BY : SARA LOMBARDO**

