Exploratory Data Analysis of 7 Million Companies using Python

A comprehensive analysis using Pandas, Matplotlib, Seaborn, and Plotly

Pankaj Thakur · Follow

Published in Jovian

9 min read · Jan 30, 2022

Photo by Floriane Vita on Unsplash

Table Of Contents

What is Exploratory Data Analysis?

Exploratory data analysis (EDA for short) is what data analysts do with large sets of data, looking for patterns and summarizing the dataset's main characteristics beyond what they learn from modeling and hypothesis testing. EDA is a philosophy that allows data analysts to approach a database without assumptions. When a data analyst employs EDA, it's like they're asking the data to tell them what they don't know.

It is an approach to data analysis, that uses these techniques:

- Maximize insights into a dataset.
- Uncover underlying structures.
- Extract important variables.
- Detect outliers and anomalies.
- Test underlying assumptions.
- Determine optimal factor settings

In this blog, we will perform exploratory data analysis on a "Company Dataset" from Kaggle.

Outline of Project:

> ***1. Download the dataset and Import all the libraries***

> *2. Data preparation and cleaning*
>
> *3. Asking and Answering questions about the data*
>
> *4. Exploratory analysis & visualization*
>
> *5. Inferences and Conclusions*

Here's my jovian link to this project, we can execute the code by running this jovian notebook

pankajthakur3999/eda-company-datasets - Jovian

Collaborate with pankajthakur3999 on eda-company-datasets notebook.

jovian.ai

1. Download the dataset and Import all the libraries

Here, In this section, we will download the dataset from Kaggle and import all the required libraries to perform analysis and visualization.

This dataset is available on Kaggle. It contains information about the 7 million companies around the world. It includes the information about the companies in which year it is established, employees status, countries, and cities where these companies are spread. We will analyze this dataset and draw some conclusions.

Dataset Link — https://www.kaggle.com/peopledatalabssf/free-7-million-company-dataset

Downloading the Dataset

Let's download the data into the Jupyter notebook. We'll use the open datasets library `opendatasets` from Jovian. Let's install and import it, and use the download method.

```
# Install opendatsets library to downlaod the data from kaggle by using link
!pip install opendatasets --upgrade --quiet
import opendatasets as od

# Dataset URL
datasets_url = 'https://www.kaggle.com/peopledatalabssf/free-7-million-compar

# Downloading the dataset
od.download(datasets_url)
```

```
Skipping, found downloaded files in "./free-7-million-company-dataset" (use
force=True to force download)
```

```
# Convert dataset into CSV file
datasets_url_to_csv = '/content/free-7-million-company-dataset/companies_sort
```

Hosted on Jovian View File

When you run the above cell, it will ask you to input your Kaggle username and Kaggle API key, which you can get from your Kaggle account go to your profile section click on the *account settings* and you will see your username and scroll a bit down you will find API key, that you can use to download the dataset.

Let's Import and Install all the required libraries:

```
!pip install jovian pandas-profiling numpy plotly --upgrade --quiet
!pip install pyyaml==5.4.1 --quiet
!pip3 uninstall statsmodels -y --quiet
!pip3 install statsmodels==0.10.0rc2 --pre --user --quiet
```

```
# Python Data Analysis Library
import pandas as pd

# NumPy is the fundamental package for scientific computing in Python
import numpy as np

# Matplotlib is a plotting library for the Python programming language and it
import matplotlib.pyplot as plt

# It sets the backend of matplotlib to the 'inline' backend
%matplotlib inline

# Seaborn is a library for making statistical graphics in Python.
import seaborn as sns

# Plotly Express is a new high-level Python visualization library
import plotly.express as px

#The plotly.figure_factory module contains dedicated functions for creating v
import plotly.figure_factory as ff
```

```
/usr/local/lib/python3.7/dist-packages/distributed/config.py:20:
YAMLLoadWarning: calling yaml.load() without Loader=... is deprecated, as
the default Loader is unsafe. Please read https://msg.pyyaml.org/load for
full details.
  defaults = yaml.load(f)
```

Hosted on Jovian View File

2. Data preparation and cleaning

Here, In this section we will perform data cleaning by using the following steps:

- Load the dataset into a data frame using Pandas
- Explore the number of rows & columns, ranges of values, etc.
- Handle missing, incorrect, and invalid data of the missing values or drop the values

Loading Dataset

Loading datasets using `DataFrame.read_csv()` a Pandas function

```
# Load the datasets using pandas
companies_datasets = pd.read_csv(datasets_url_to_csv)

# Get the dataframe
companies_datasets.head()
```

	Unnamed: 0	name	domain	year founded	industry	size range	locality	country	
0	5872184	ibm	ibm.com	1911.0	information technology and services	10001+	new york, new york, united states	united states	
1	4425416	tata consultancy services	tcs.com	1968.0	information technology and services	10001+	bombay, maharashtra, india	india	
2	21074	accenture	accenture.com	1989.0	information technology and services	10001+	dublin, dublin, ireland	ireland	linl
3	2309813	us army	goarmy.com	1800.0	military	10001+	alexandria, virginia, united states	united states	li
4	1558607	ey	ey.com	1989.0	accounting	10001+	london, greater london, united kingdom	united kingdom	linkedin

Hosted on Jovian View File

The `Dataframe.info()` method prints information about the DataFrame. The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values).

```
# Check the datasets info
companies_datasets.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7173426 entries, 0 to 7173425
Data columns (total 11 columns):
 #   Column                    Dtype
---  ------                    -----
 0   Unnamed: 0                int64
 1   name                      object
 2   domain                    object
 3   year founded              float64
 4   industry                  object
 5   size range                object
 6   locality                  object
 7   country                   object
 8   linkedin url              object
 9   current employee estimate  int64
 10  total employee estimate    int64
dtypes: float64(1), int64(3), object(7)
memory usage: 602.0+ MB
```

Hosted on Jovian View File

Here we can see

- The data type of each column which is int64, object, and float
- The number of rows is 7173426.
- The number of columns is 11
- The file size after loading the dataset was 602.0+ MB.

Now we have a rough idea of our dataset and we can further process for analysis

The `DataFrame.describe()` the function is used to generate descriptive statistics that summarize the central tendency, dispersion, and shape of a dataset's distribution, excluding NaN values. It is used to view some basic statistical details like percentile, mean, std, etc.

```
# Description of datasets
companies_datasets.describe()
```

	Unnamed: 0	year founded	current employee estimate	total employee estimate
count	7.173426e+06	3.566446e+06	7.173426e+06	7.173426e+06
mean	3.586712e+06	2.001747e+03	1.387121e+01	3.225438e+01
std	2.070790e+06	2.096627e+01	3.545919e+02	8.741556e+02
min	0.000000e+00	1.451000e+03	0.000000e+00	1.000000e+00
25%	1.793356e+06	1.999000e+03	1.000000e+00	1.000000e+00
50%	3.586712e+06	2.009000e+03	1.000000e+00	2.000000e+00
75%	5.380069e+06	2.013000e+03	4.000000e+00	7.000000e+00
max	7.173425e+06	2.103000e+03	2.740470e+05	7.169060e+05

Hosted on Jovian View File

Let's drop some columns which are not required for our analysis

```
# Check all the columns of dataset
companies_datasets.columns
```

```
Index(['Unnamed: 0', 'name', 'domain', 'year founded', 'industry',
       'size range', 'locality', 'country', 'linkedin url',
       'current employee estimate', 'total employee estimate'],
      dtype='object')
```

```
# Let's drop the columns which are not of our use (Unnamed: 0, domain, linked
new_datasets = companies_datasets.drop(['Unnamed: 0','domain','size range', '

# Get the dataframe
new_datasets.head(3)
```

	name	year founded	industry	locality	country	current employee estimate	total employee estimate
0	ibm	1911.0	information technology and services	new york, new york, united states	united states	274047	716906
1	tata consultancy services	1968.0	information technology and services	bombay, maharashtra, india	india	190771	341369
2	accenture	1989.0	information technology and services	dublin, dublin, ireland	ireland	190689	455768

Hosted on Jovian View File

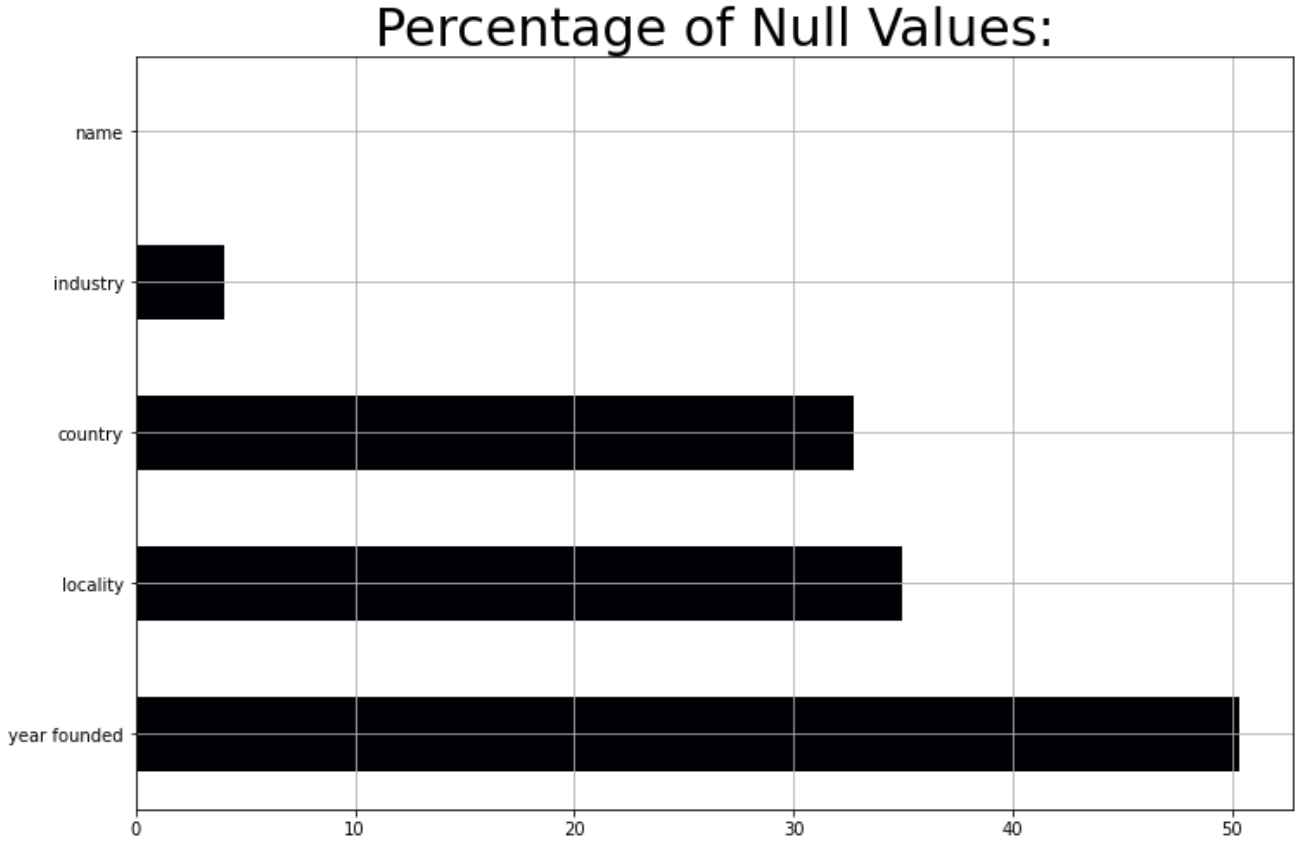
We have dropped the column name `Unnamed: 0 domain size range linkedin url` which was not required for our analysis.

Now, let's calculate the null percentages of the columns that we have for analysis.

```
null_perceentages = new_datasets.isna().sum().sort_values(ascending=False) /
null_perceentages
```

```
year founded                 50.282529
locality                     34.973874
country                      32.748745
industry                      4.042741
name                          0.000042
total employee estimate       0.000000
current employee estimate     0.000000
dtype: float64
```

```
# Plot a bar chat to see the missing peercentage
plt.title("Percentage of Null Values:" , fontsize=30)
null_perceentages[null_perceentages != 0].plot(figsize=(12,8),kind='barh',gri
```



Hosted on Jovian View File

Here we can see `year founded` is missing around 50% of the data and the `locality` has 34% of the missing data. So let's drop all the missing years from the datasets and perform some analysis on the column `year founded`

3. Asking and Answering questions about the data

Let's answer a few questions about the data. Here we will do data manipulations and visualizations while answering the questions.

> ***We will try to answer the following questions about the data:***

1. How is the distribution of companies been over the years?
2. How are the different industries are scattered in that country?
3. How are the industries scattered in different countries over the year?
4. Which company has the most spread around the world?
5. What are the top 15 countries where the companies are higher?
6. What are the top 10 industries around the world?
7. Which city has more number of IT industries with respect to the number of employees within the country?
8. Create a TreeMap for the country, state, and city

Before proceeding further, We will make a few changes to our dataset:

Let's rename a few columns, convert the items of the column in the title case, and then we will make a copy of the dataset

```
# Make some changes in dataframe

# Renaming the columns
new_datasets = new_datasets.rename(columns={"year founded": "year_founded","c

# Convert strings in the Series/Index to titlecase.
new_datasets.country = new_datasets.country.str.title()
new_datasets.industry = new_datasets.industry.str.title()
```

Hosted on Jovian View File

Let's make a copy of the dataset by `dataframe.copy()`

```
# Let's make a copy
copied_dataset = new_datasets.copy()

# Drop all the Null Values
copied_df = copied_dataset.dropna()

# Convert the data data into CSV file
copied_df.to_csv('copied_df.csv')

# Let's take sample of 10000 rows to perform analysis
copied_df_sample = pd.read_csv('copied_df.csv',index_col = 0, nrows=10000)
```

Hosted on Jovian View File

Let's take 10,000 rows of data as sample data.

So, now the question is why are we taking the sample of data?

Sampling is very important and useful with data sets that are too large to efficiently analyze in full. Identifying and analyzing a representative sample is more efficient and cost-effective than dealing with the entirety of the data.

Here our dataset is too large as it contains `7173426` i.e 7+ million rows of data. So that's why we are taking the sample of 10,000 rows of data.

So, now we will make a copy of our original data `new_datasets` and then take a sample of 10,000 rows.

4. Exploratory Analysis & Visualization

Here in this section, we will perform the visualizations by exploring the data and getting some insights by answering the following questions.

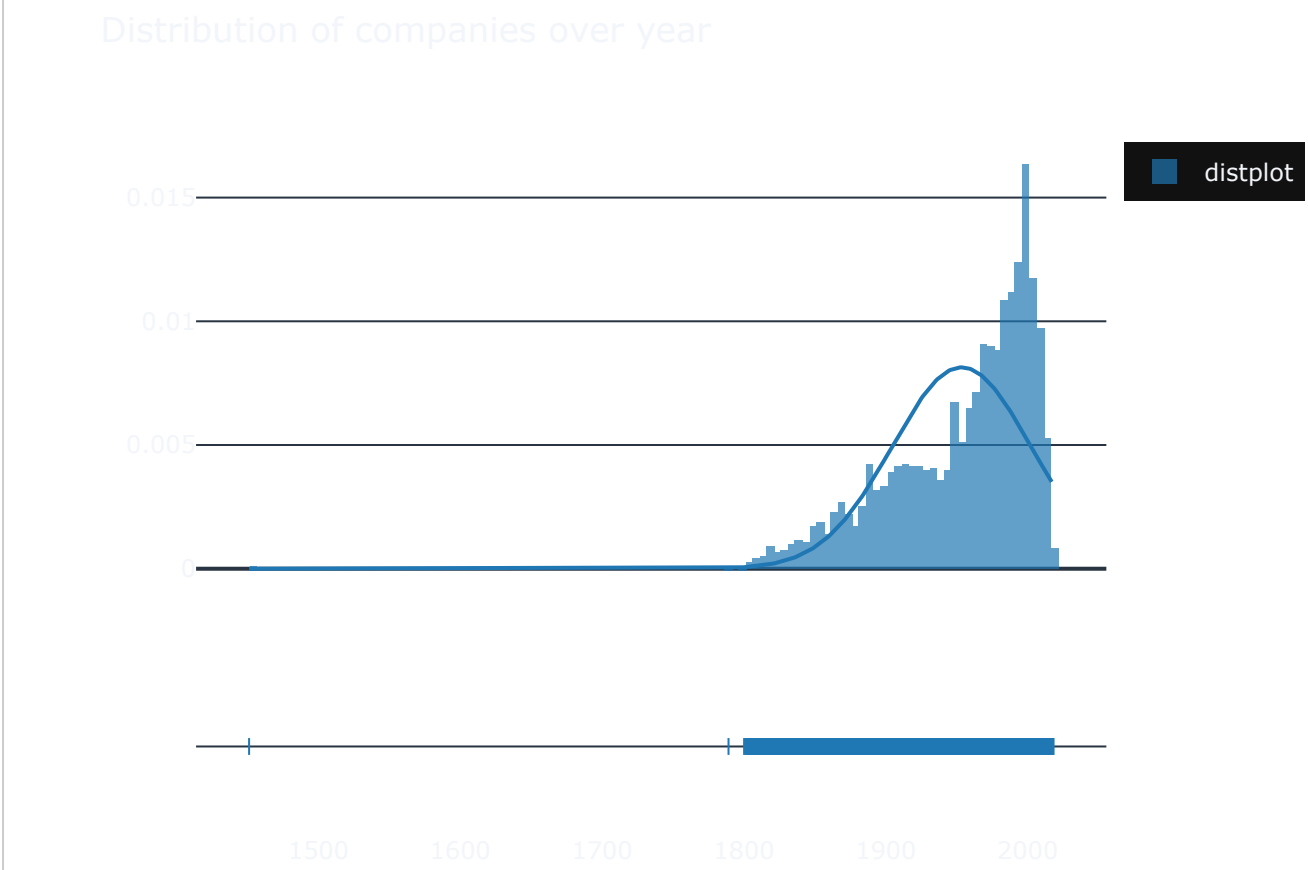
> 1. How is the distribution of companies been over the years?

To get an idea of what our companies distribution looks like over the years we need to plot a histogram.

- A histogram is the most commonly used graph to show frequency distributions.

- A frequency distribution shows how often each different value in a set of data occurs
- It helps us to see the shape of the data's distribution, especially when determining whether the output of a process is distributed approximately normally.
- It helps us to see whether a process change has occurred from one time period to another.

```
# Let's plot histogram
x = copied_df_sample["year_founded"]
hist_data = [x]
group_labels = ['distplot']

fig = ff.create_distplot(hist_data, group_labels,curve_type = 'normal',bin_si
fig.update_layout(template="plotly_dark",title='Distribution of companies ove

fig.show()
```



Hosted on Jovian View File

Insights from the above chart

- Here we can see from the chart, that most of the countries are established around 1946 to 2006 around the world.
- Our chart is showing left-skewed distribution.
- A “skewed left” distribution is one in which the tail is on the left side.
- If the histogram is skewed left, the mean is less than the median. Here our mean is somewhere around 1953 and the median is 1968.
- It is because skewed-left data(year_founded) have a few small values that drive the mean downward but do not affect where the exact middle of the data is (that is, the median)

2. How are the different industries are scattered in a country?

Here we will plot a scatter plot that helps us to compare the different industries in a country.

Why are scatter plots so important?

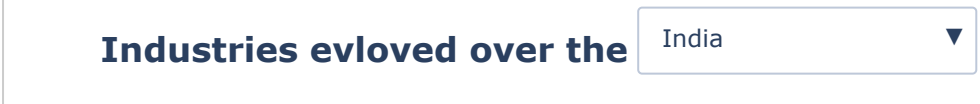
- Scatter plots’ primary uses are to observe and show relationships between two numeric variables.
- It is useful for identifying patterns in data.

But in our case, only the y-axis is numeric(year from 1800).

So how do read the below chart?

To read the below chart click on the dropdown of the country(left corner of the chart). It will show you how that industry has grown in that country from the year 1800 to 2016.

```
import os
import plotly.graph_objects as go
copied_df_sample = copied_df_sample[copied_df_sample.year_founded > 1800]

earth = sorted(set(copied_df_sample["country"]))

fig=go.Figure()

region_plot_names = []
buttons=[]

default_state = "India"

for region_name in earth:
  region = copied_df_sample[(copied_df_sample["country"]==region_name)]
  fig.add_trace(go.Scatter(x=region["industry"], y=region["year_founded"], li
  region_plot_names.extend([region_name])

for region_name in earth:
    buttons.append(dict(method='update',
                        label=region_name,
                        args = [{'visible': [region_name==r for r in region_p

# Add dropdown menus to the figure
fig.update_layout(showlegend=False, updatemenus=[{"buttons": buttons, "direct




fig.update_layout(template="plotly",title='<b>Industries evloved over the yea
                  yaxis_title='<b>Year from 1800 to 2016</b>',
                  xaxis_title="<b>Industry</b>")

fig.show()
```



Insights from the above scatter plot

- For example, click on the `United States` from the dropdown. It will show the trend in different industries. the `military` has started at the earliest(i.e 1800) but the `e-learning` has started around 1918.

3. How are the industries scattered in different countries over the years?

Here we will plot a scatter plot which helps us to compare the industries in different countries.

```
copied_df_sample = copied_df_sample[copied_df_sample.year_founded > 1800]

Industry = sorted(set(copied_df_sample["industry"]))

fig=go.Figure()

region_plot_names = []
buttons=[]

default_state = "Insurance"

for region_name in Industry:
  region = copied_df_sample[(copied_df_sample["industry"]==region_name)]
  fig.add_trace(go.Scatter(x=region["country"], y=region["year_founded"], lir
  region_plot_names.extend([region_name])

for region_name in Industry:
    buttons.append(dict(method='update',
                        label=region_name,
                        args = [{'visible': [region_name==r for r in region_p

# Add dropdown menus to the figure
fig.update_layout(showlegend=False, updatemenus=[{"buttons": buttons, "direct




fig.update_layout(template="plotly",title='<b>Industries evloved over the yea
                  yaxis_title='<b>Year from 1800 to 2016</b>',
                  xaxis_title="<b>Industry</b>")

fig.show()
```

Industries evloved over the years

Insights from the above chart

- For example, click on the `Financial Services` from the dropdown of industry. You will see this in `United States` it is started in the early 1800 but in `India` it is started around 1887.

Before proceeding further let's drop the column `year_founded` from the `new_datasets` and then drop all the null values and perform the analysis by taking the sample.

```
# Let's drop the column year_founded
data_drop_year = new_datasets.drop(['year_founded'], axis =1)

# Drop all the null values
Data = data_drop_year.dropna()

# Convert the dataframe to CSV
Data.to_csv('Data.csv')

# Let's take a sample of 10,000 rows
sample_df = pd.read_csv('Data.csv',index_col = 0, nrows=10000)
```

Hosted on Jovian View File

4. Which company has the most spread around the world?

Here we will plot a word cloud on the column `name` to get insights about the companies having more spread.

What is a word cloud?

A word cloud is a collection, or cluster, of words depicted in different sizes. The bigger and bolder the word appears, the more often it's mentioned within a given text and the more important it is.

```
# Generate a word cloud image to check the spread of diiferent company
from PIL import Image
from wordcloud import WordCloud
text = sample_df['name'].values
wordcloud = WordCloud(width=800, height=400, background_color='white',).gener
# Display the generated image:
# the matplotlib way:


plt.figure( figsize=(25,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show();
```

Hosted on Jovian View File

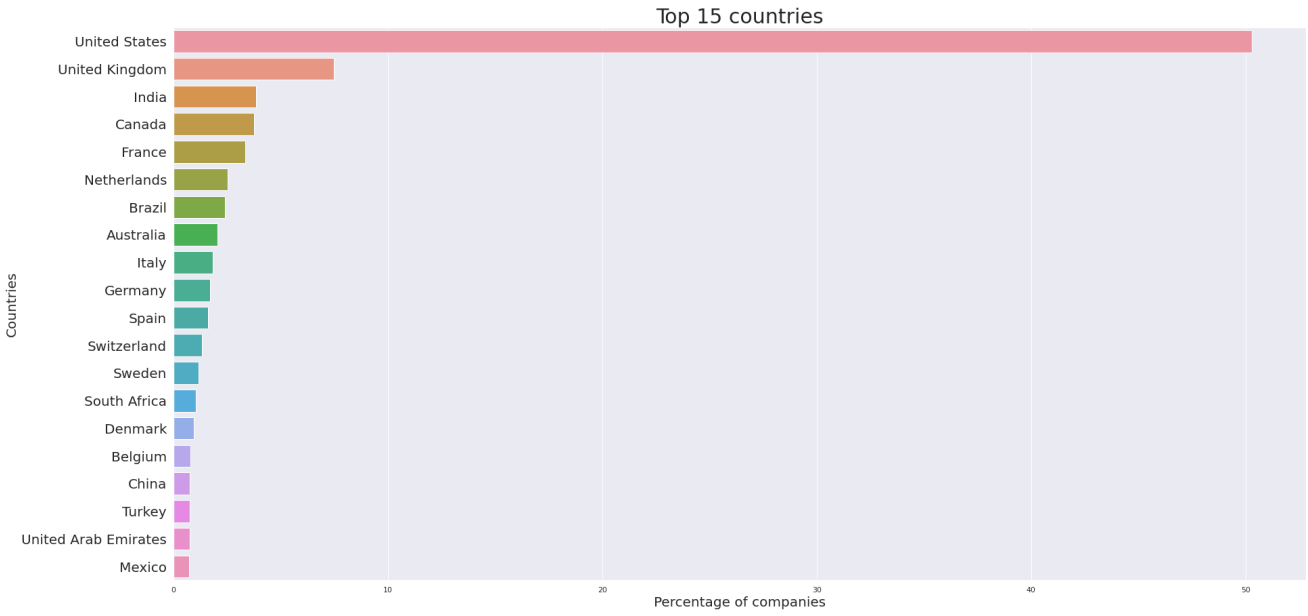
From the above word cloud, we can analyze that IBM, tata, Accenture is having more companies.

5. What are the top 15 countries where the companies are higher?

Here we will plot a bar chart on the column `country` to get the top 15 countries.

```
country_percentage = sample_df['country'].value_counts() * 100/len(sample_df)
sns.set(style="darkgrid")

# Counttry_count
plt.figure(figsize=(30,15))
plt.yticks(fontsize=20)
sns.barplot(country_percentage.values[:20], country_percentage.index[:20])

plt.title('Top 15 countries',fontsize=30)
plt.ylabel('Countries', fontsize=20)
plt.xlabel('Percentage of companies',fontsize=20)

plt.show();
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning:

Pass the following variables as keyword args: x, y. From version 0.12, the
only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
```



Insights from the above chart

- Us is contributing to 53% of all the 43 countries
- The second highest is Uk which is contributing nearly 8 %
- India which is 4th highest contributing to 4 % of total companies in the world

Why China is in the 17th position?

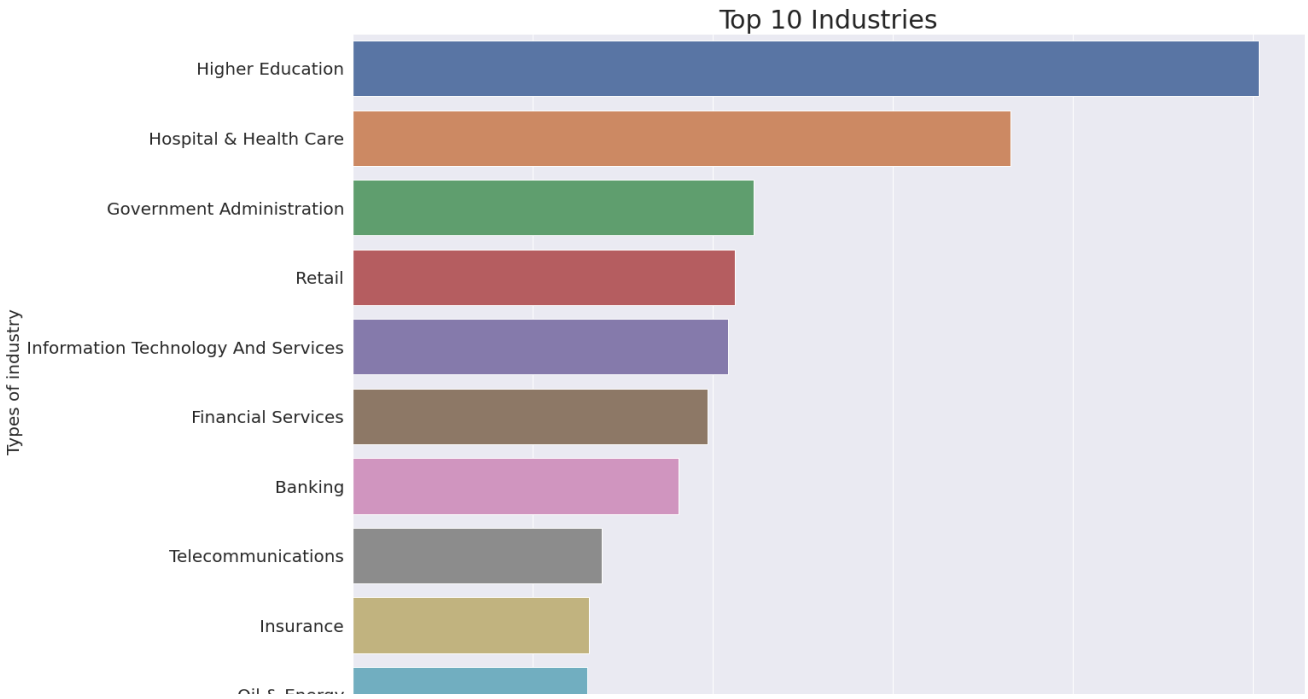
Answer — It is because our dataset does not contain the companies information about china.

6. What are the top 10 industries around the world?

Here, we will analyze the column `industry` to get the top 10 industries around the world.

```
industry_percentage = sample_df['industry'].value_counts() * 100/len(sample_d
sns.set(style="darkgrid")

# industry_count
plt.figure(figsize=(20,15))
plt.yticks(fontsize=20)
sns.barplot(industry_percentage.values[:10], industry_percentage.index[:10])

plt.title('Top 10 Industries',fontsize=30)
plt.ylabel('Types of industry', fontsize=20)
plt.xlabel('Percentage of companies',fontsize=20)
plt.show();
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning:

Pass the following variables as keyword args: x, y. From version 0.12, the
only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
```



Insights from the above chart

- From the chart, we can see the higher education sector is contributing nearly 10% which is the highest followed by hospital and health care which is nearly 7%
- Insurance is contributing around 3 % all over the world.

7. Which city has more number of IT industries in terms of employees within the country?

Here we will select a few items from the column `industry` to create a new data frame that contributes to the IT industry in different cities of a country.

To get this we will need to do the following things.

1. Select the IT industry from the column `industry`
2. Split the `locality` column to get the city and state name and the plot the chart

1. Select the IT industry from the column `industry`

Not only do information and technology contribute to IT industries So, Let's get a full list of industries and manually select those related to IT (my selection is not very critical and is subjective)

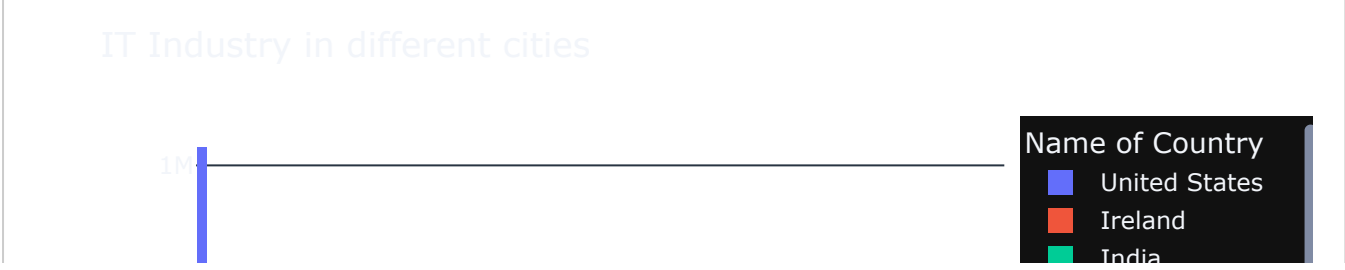
Let's get selected industry contributing to the IT industry

```
# Selected industries whcich is contributin to IT industries
IT_industries = [
 'Animation',
 'Biotechnology',
 'Computer & Network Security',
 'Computer Games',
 'Computer Hardware',
 'Computer Networking',
 'Computer Software',
 'Consumer Electronics',
 'Defense & Space',
 'E-Learning',
 'Industrial Automation',
 'Information Services',
 'Information Technology And Services',
 'Internet',
 'Mechanical Or Industrial Engineering',
 'Program Development',
 'Telecommunications',
 'Wireless'
]
```

Hosted on Jovian View File

2. Split the `locality` column to get the city and state name and plot the chart

```
# Let's split the column locality to get the city name and state name in two
sample_df[['city','state','country']] = sample_df.locality.apply(lambda x: pd

# Convert strings in the Series/Index to titlecase.
sample_df.country = sample_df.country.str.title()
sample_df.city = sample_df.city.str.title()
sample_df.state = sample_df.state.str.title()

# We are only interested in companies, where at least one current employee is
df_companies = sample_df[sample_df['current_employee_estimate'] > 0]

# Crate a dataframe for the IT industries
it_frame = df_companies[df_companies.industry.isin(IT_industries)]

# Let's plot a Bar plot
it_df = it_frame.copy()
it_df = it_df.sort_values('total_employee_estimate', ascending=False)

it_df = it_df[:100]

fig = px.histogram(it_df, x='city', y='total_employee_estimate',

                           color='country',
                           hover_name='city')

fig.update_layout(template='plotly_dark',
                  title="IT Industry in different cities",
                  yaxis_title="Total out of 100",
                  xaxis_title="Cities around the world",
                  legend_title ='Name of Country')
fig.show()
```



From the chart, we can say India's IT industry is evolved around the cities Banglore which is highest followed by Bombay, Noida, Pune, Hyderabad, and Gurgaon.

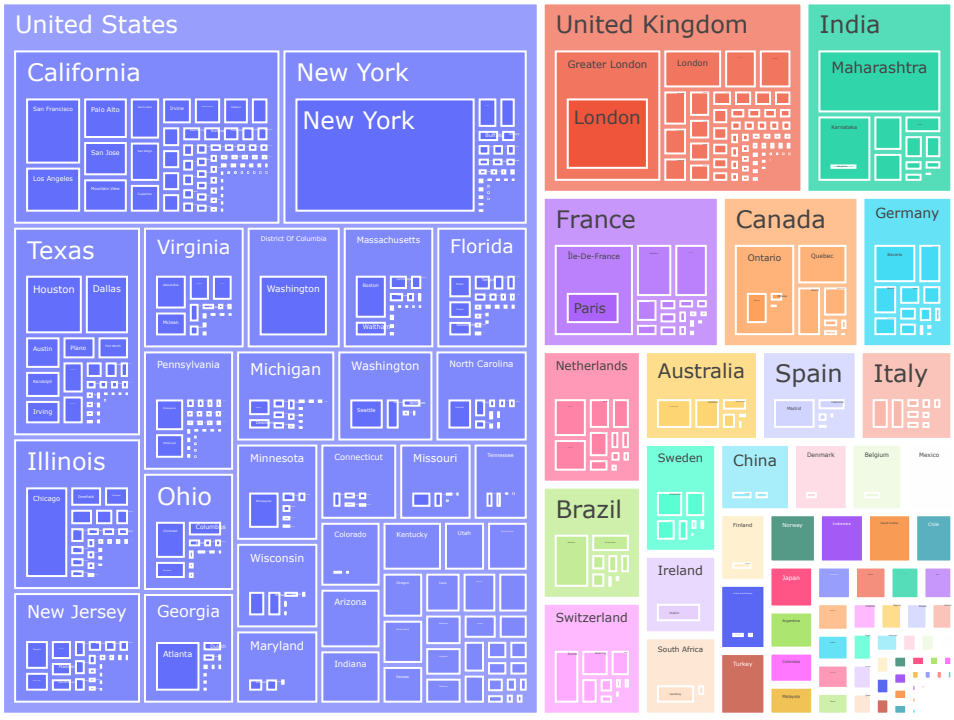
8. Create a TreeMap for the country, state, and city

What is a treemap?

- Treemaps display hierarchical (tree-structured) data as a set of nested rectangles. Each branch of the tree is given a rectangle, which is then tiled with smaller rectangles representing sub-branches. A leaf node's rectangle has an area proportional to a specified dimension of the data.[1] Often the leaf nodes are colored to show a separate dimension of the data.

- Our dataset contains the columns; country, state, and city to see the visuals of the treemap.

```
# Plot a treemap
fig = px.treemap(sample_df,
                 path=["country","state","city"],
                 values= "current_employee_estimate",
                 color_continuous_scale="RdBu")
fig.show()
```



Hosted on Jovian View File

- By clicking on each country, it will show the state of that country, and again by clicking on that state it will show the city.

- 60% of the data is representing the United States in terms of their current employee estimate

Conclusion

- Most countries are established around 1946 to 2006 around the world.
- Most of the companies are established in the United States followed by UK and India.
- We can analyze that IBM, tata, Accenture are having more companies. Us is contributing to 53% of all the 43 countries
- The second highest is Uk which is contributing nearly 8 %
- India which is 4th highest contributing to 4 % of total companies in the world
- We have data limitations regarding china It does not contain more companies' information about china.
- From the chart, we can see the higher education sector is contributing nearly 10% which is the highest followed by hospital and health care which is nearly 7%
- Insurance is contributing around 3 % all over the world.
- From the chart, we can see in India information and technology is highest.

Future Work

- Visualize the cities where industries are located around the world
- Get the colorful map and try different columns.
- We can do stratified sampling for every country
- More analysis can be drawn out from this dataset. As we haven't used all columns in the analysis. If we can combine more columns, more interesting results will come out.
- We can also combine it with some other interesting datasets.

References

- Plotly Documentation: https://plotly.com/python/

- EDA from scratch by Aakash NS: https://www.youtube.com/watch?v=kLDTbavcmd0&t=5315s
- Another EDA video from Aakash NS: https://www.youtube.com/watch?v=B4GbWjUFUGk
- Aakash N S. Analyzing Tabular Data with Pandas, 2021. https://jovian.ai/aakashns/python-pandas-data-analysis
- Pandas Documentation: https://pandas.pydata.org/docs/

Data Analysis | Python | Visualization | Eda | Beginners Guide

Follow

Written by Pankaj Thakur

40 Followers · Writer for Jovian

Over the last 6 months, I have done 600+ hours of coursework, 10 coding assignments, and 3 projects (Web scraping, EDA, Tableau).

More from Pankaj Thakur and Jovian

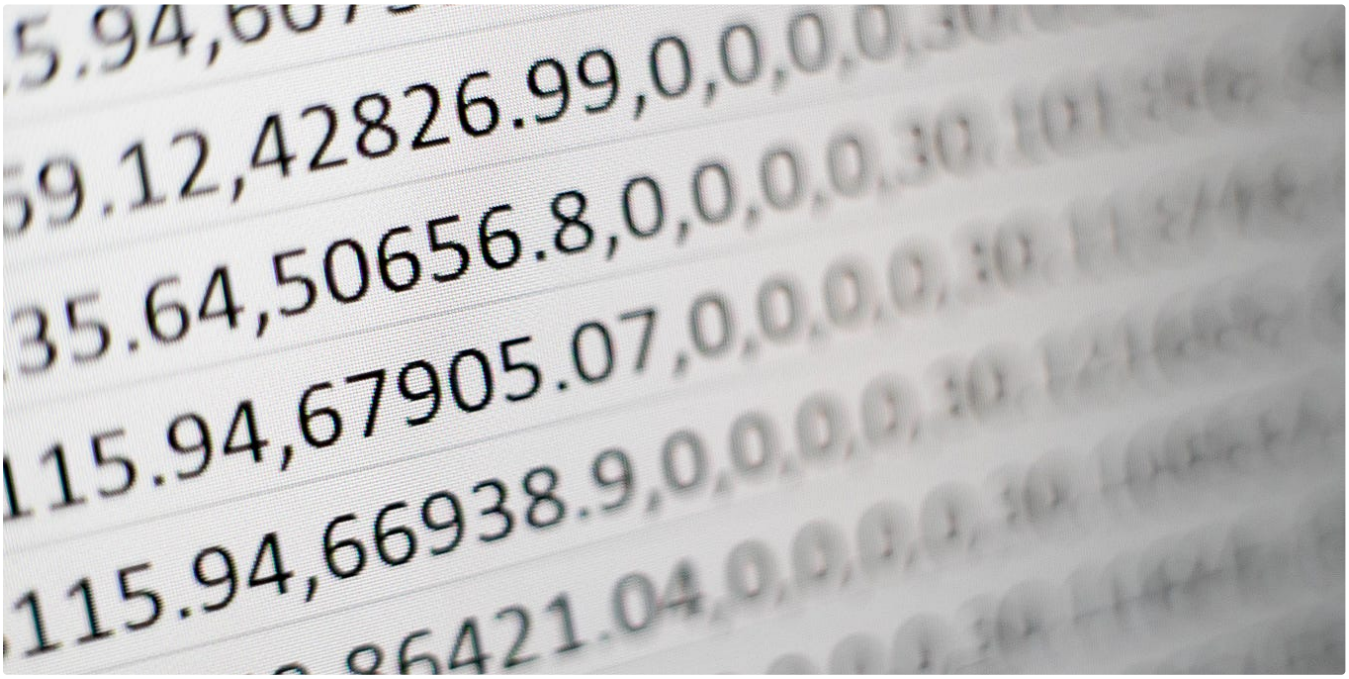
 Pankaj Thakur in Jovian

Web Scraping Top Insurance Companies using Python and BeautifulSoup

using Python's Requests, Beautiful Soup, and Pandas

5 min read · Dec 3, 2021

82



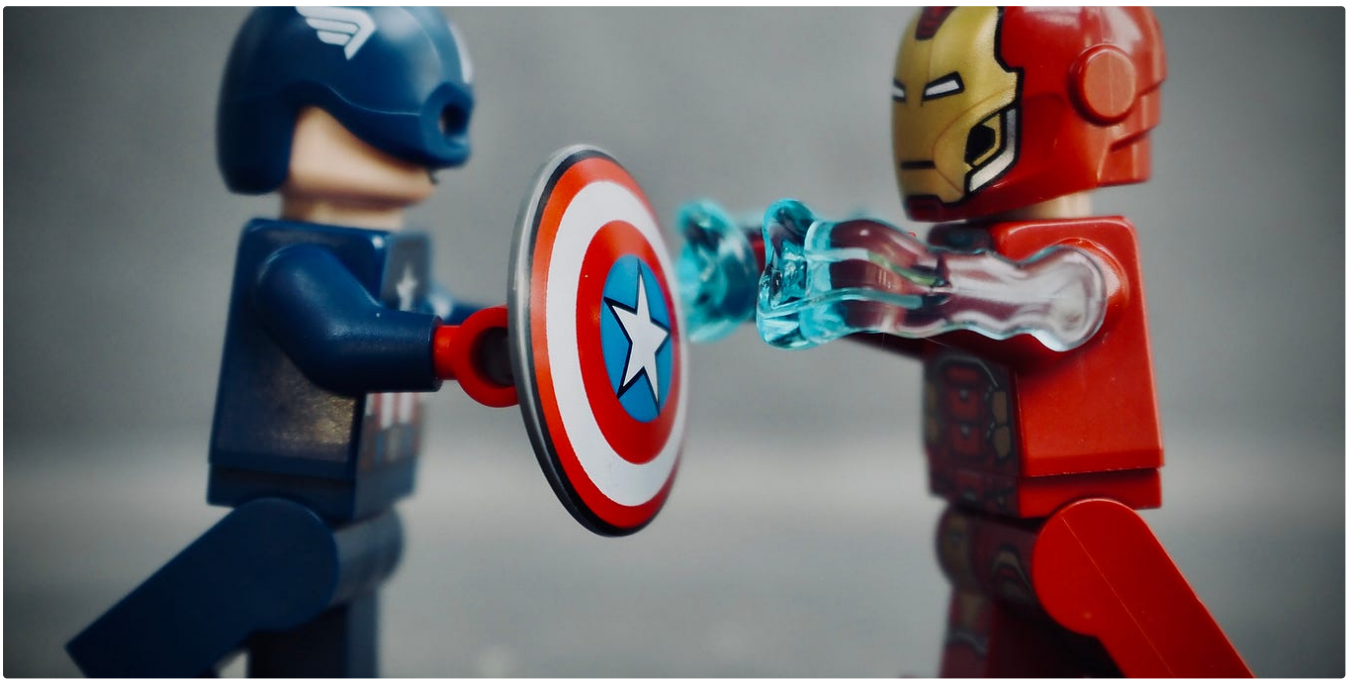
 Nelson M. in Jovian

Adding Meaning to Your Power BI KPI Reports and Dashboards

In the world of analytics, where data drives decisions, the importance of context cannot be overstated.

✦ · 3 min read · Oct 6

44 1



Nelson M. in Jovian

Power BI vs. Tableau : Pick one, Trash the other

Over the last few years, the landscape of data analytics tools has seen significant changes, with Power BI emerging as a powerful...

✦ · 3 min read · Oct 7

17 2

Pankaj Thakur in Jovian

Numpy Trigonometric Functions

Python, Numpy, Trignometry, Numpy trigonometric function

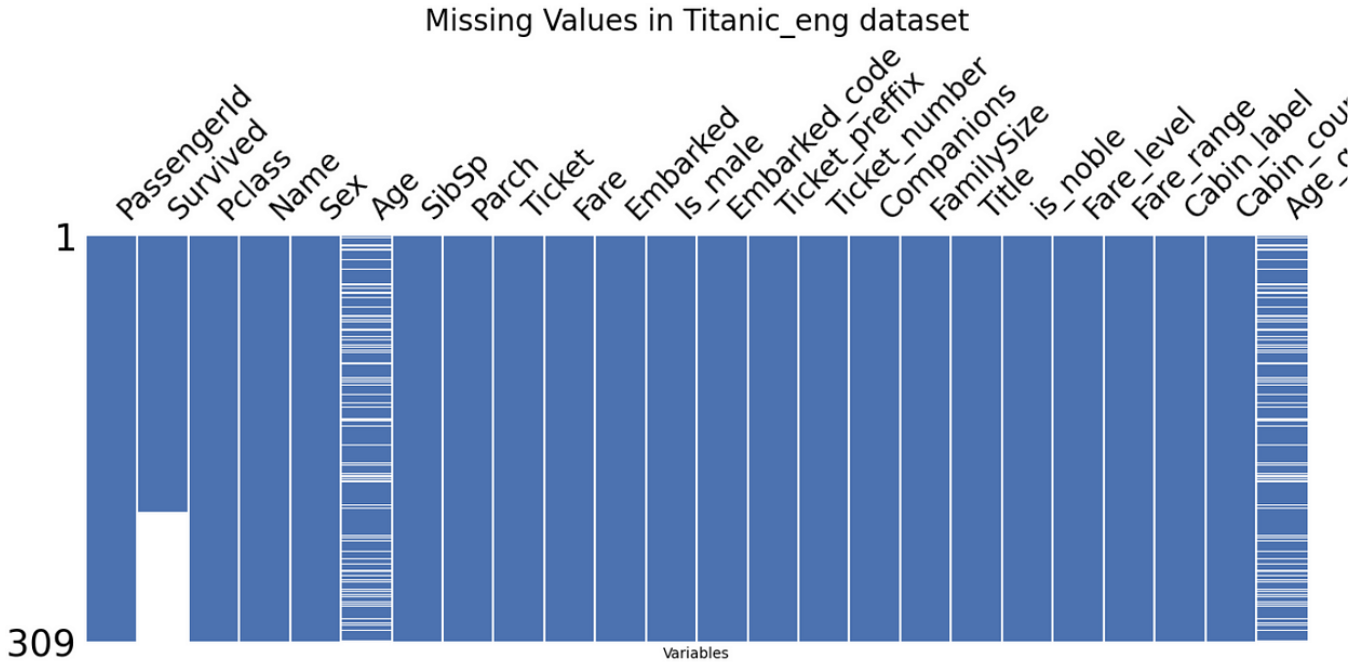
5 min read · Dec 21, 2021

55

See all from Pankaj Thakur

See all from Jovian

Recommended from Medium



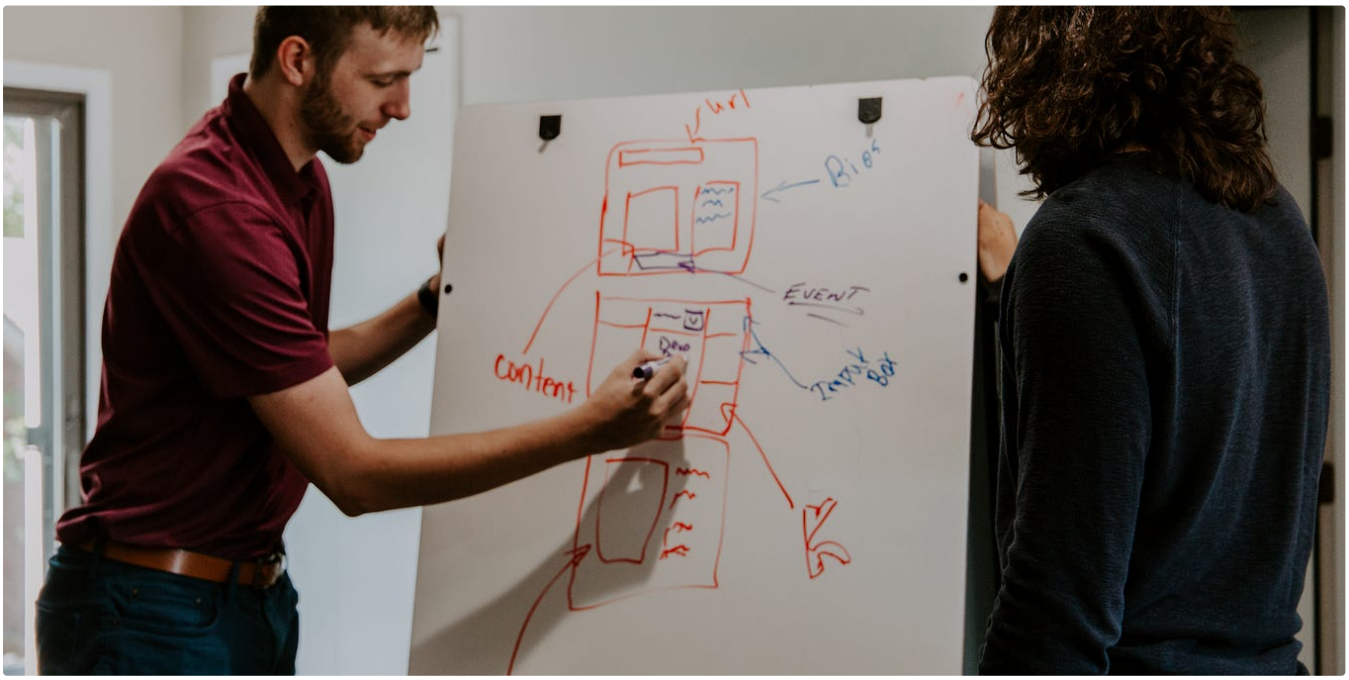
Praoiticica

Titanic—Data Cleaning and Feature Engineering

The Titanic dataset is one of the best datasets to practice data cleaning and feature engineering. This is the 1st step before applying ML

18 min read · Jul 17

16 1



Simon Benavides Pinjosovsky, PhD

Normalize data before or after split of training and testing data?

When working with machine learning models, it is important to preprocess the data before training the model. One common preprocessing...

3 min read · Jul 5

117 1

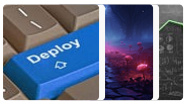
Lists

Coding & Development

11 stories · 260 saves

Practical Guides to Machine Learning

10 stories · 658 saves



Predictive Modeling w/ Python

20 stories · 574 saves

New_Reading_List

174 stories · 187 saves

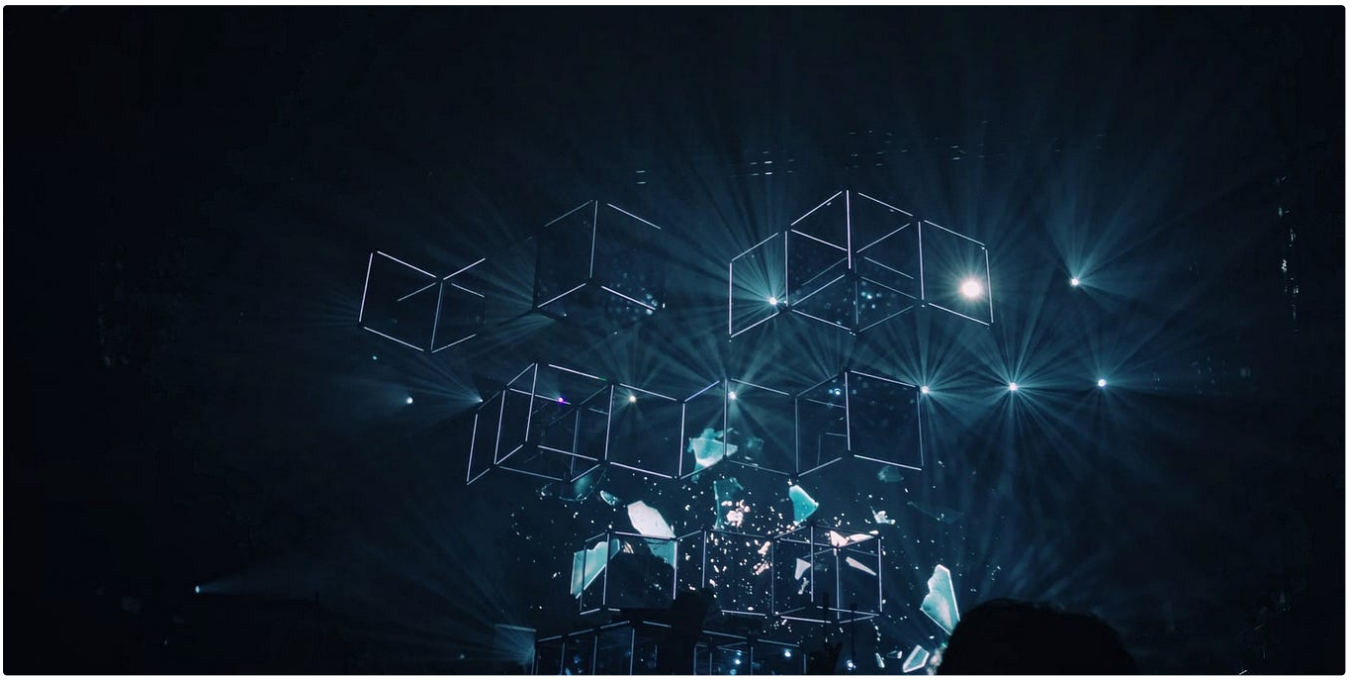
Python Programming

Data Visualization for Exploratory Data Analysis (EDA)

Data Visualization Techniques Using Python

· 6 min read · 3 days ago

28



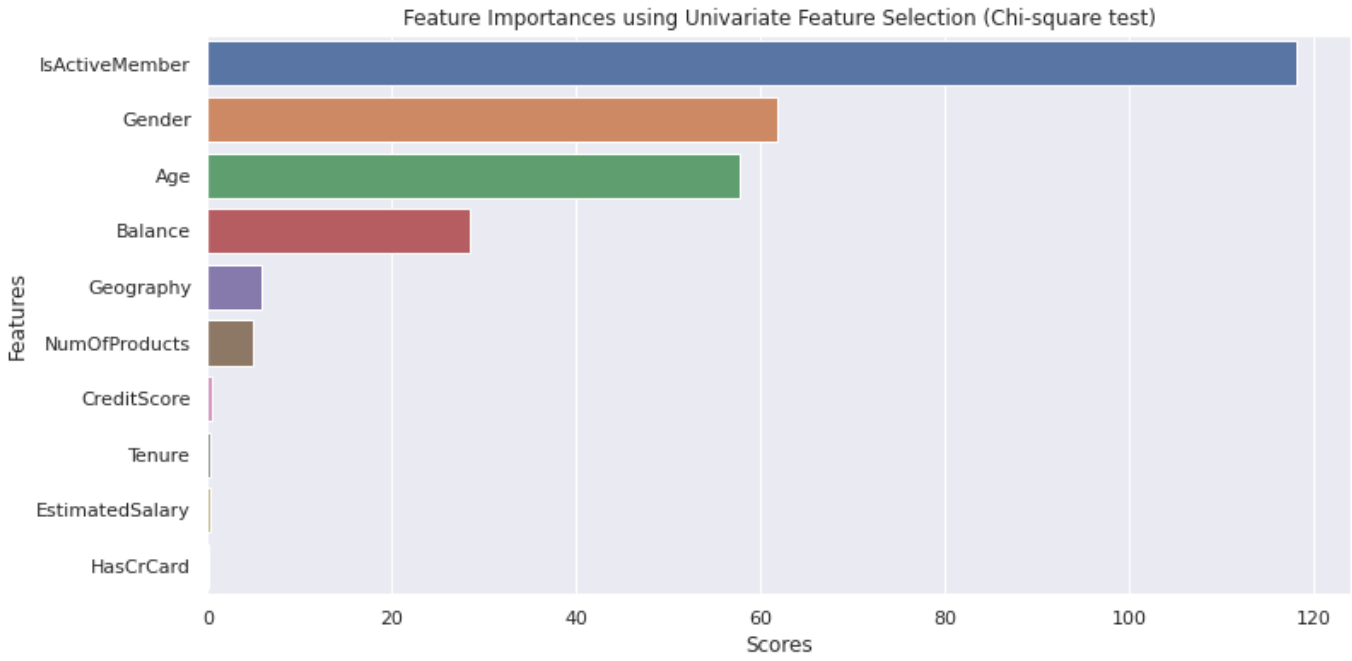
Virat Patel

I applied to 230 Data science jobs during last 2 months and this is what I've found.

A little bit about myself: I have been working as a Data Analyst for a little over 2 years. Additionally, for the past year, I have been...

· 3 min read · Aug 11

2.3K 53



daython3

Mastering the Art of Feature Selection: Python Techniques for Visualizing Feature Importance

Feature selection is one of the most crucial steps in building machine learning models. As a data scientist, I know the importance of...

6 min read · May 13

28

Mochamad Kautzar Ichramsyah in CodeX

Automate the exploratory data analysis (EDA) to understand the data faster and easier

What is EDA?

11 min read · Jul 12

1.6K 20

See more recommendations