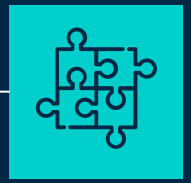
Data Analytics

HEC PARIS

Decisions in the World IP Organization on domain name dispute

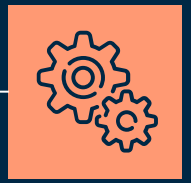
Antoine CAROFF
Denisse MONROY MONDRAGÓN
Romane-Laure PERRIN
Guillaume TERTRE

TABLE OF CONTENTS



01

ISSUES & OUR PROPOSAL



02

OUR CODE

Scraping & defining an appropriate dataframe

03

OUR ANALYSIS

Thanks to statistics

Identified issues in trademark law

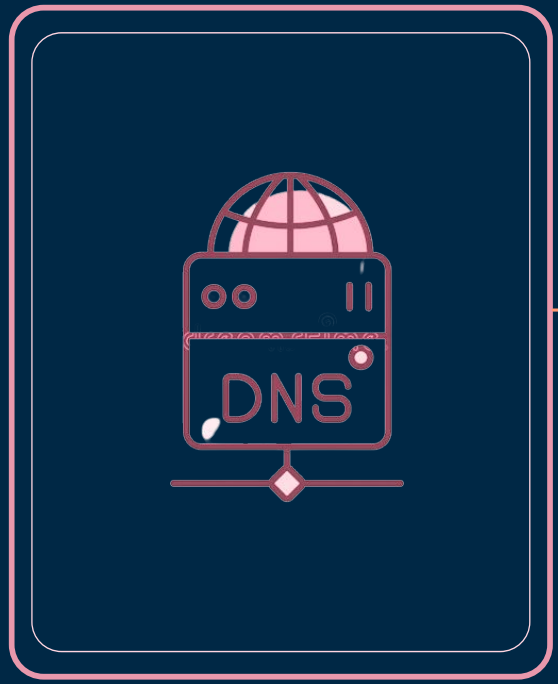
Trademark infringement and dilution

01

OUR PROPOSAL

Using Python, we want to dig deep into IP decisions regarding domain names disputes, and analyse:

- Success rate throughout the years
- Success rate per sector
- Success rate per panelist

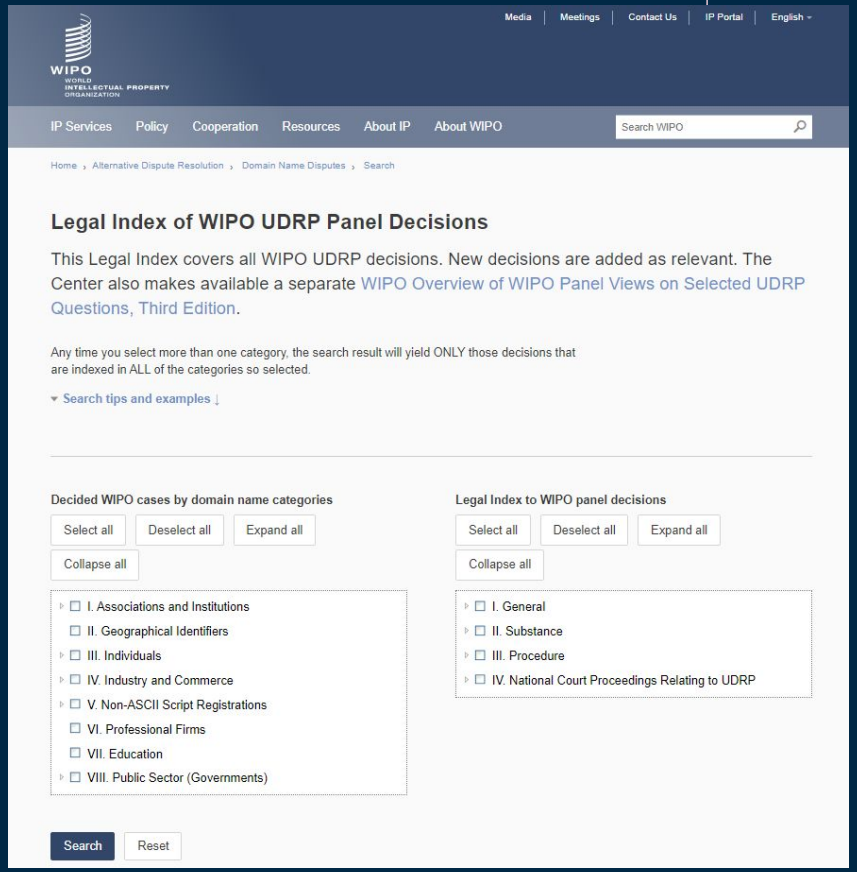


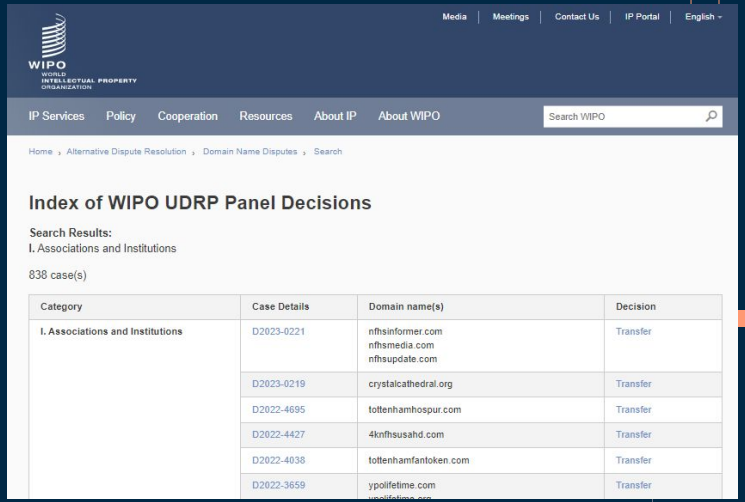
/02

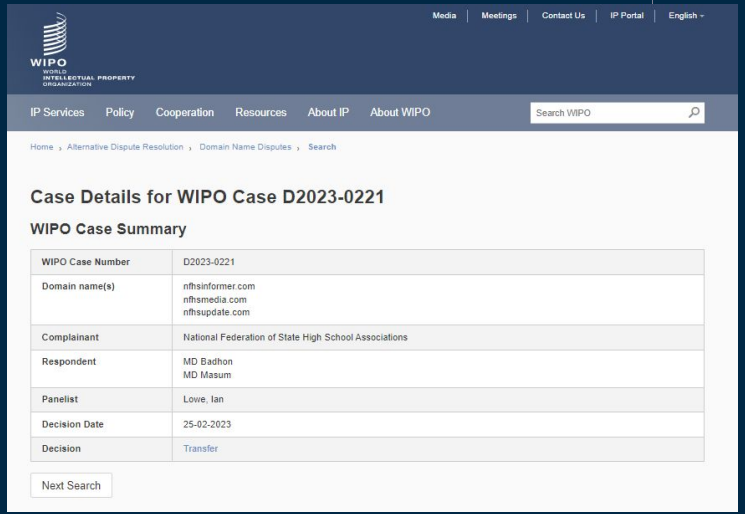
/OUR CODE

What were the steps of our code, what were the choices we made and why?

Website's structure







SCRAPING : getting the case details for each dispute

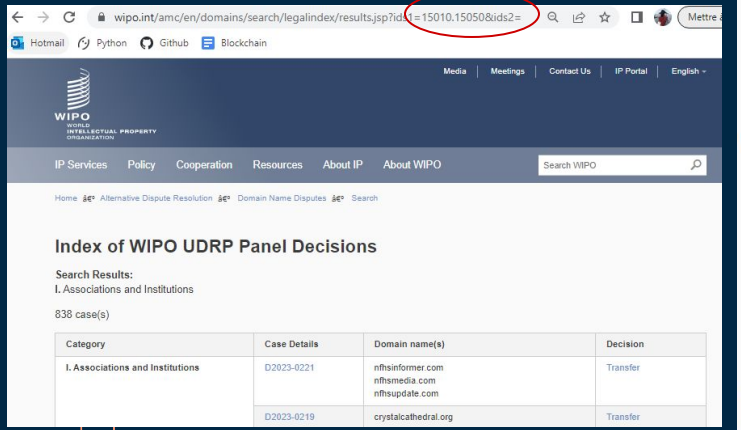
```
Microsoft Windows [Version 10.0.19044.2728]
(c) Microsoft Corporation. All rights reserved.

C:\Users\denis>ipython
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.10.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import re
   ...: from selenium.webdriver.common.by import By
   ...: import os
   ...: from bs4 import BeautifulSoup
   ...: from collections import Counter, defaultdict
   ...: import pandas as pd
   ...: import time
   ...: import numpy as np
   ...: from selenium import webdriver
   ...: from selenium.webdriver.chrome.service import Service
   ...: from webdriver_manager.chrome import ChromeDriverManager
   ...: from selenium.webdriver.common.keys import Keys as KeysBrowser

In [2]: driver = webdriver.Chrome(service=Service(ChromeDriverManager().install())) # Opening browser
   ...: driver.get("https://www.wipo.int/amc/en/domains/search/legalindex/")  # Get to WIPOâs main page

DevTools listening on ws://127.0.0.1:54300/devtools/browser/82e6e68c-3062-44d6-9517-b8f7c7b76b43

In [3]: page_source = "https://www.wipo.int/amc/en/domains/search/legalindex/"

In [4]: soup = BeautifulSoup(driver.page_source)

In [5]: lis = soup.find_all("li", id=re.compile("^fancytree-Cb1-"))

In [6]: del lis[3]

In [7]: DFS = []

In [8]: links=[]

In [9]: for li in lis:
   ...:     url = "https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=" + li.get("id").split("-"
   ...: )[-1]
   ...:     print(url)
   ...:     links.append(url)
   ...:
https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15010.15050
https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15050
https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15060.15130
https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15310.15410
https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15410
https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15420
https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15430.15500

In [10]: links
Out[10]:
['https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15010.15050',
 'https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15050',
 'https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15060.15130',
 'https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15310.15410',
 'https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15410',
 'https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15420',
 'https://www.wipo.int/amc/en/domains/search/legalindex/results.jsp?ids1=15430.15500']
```

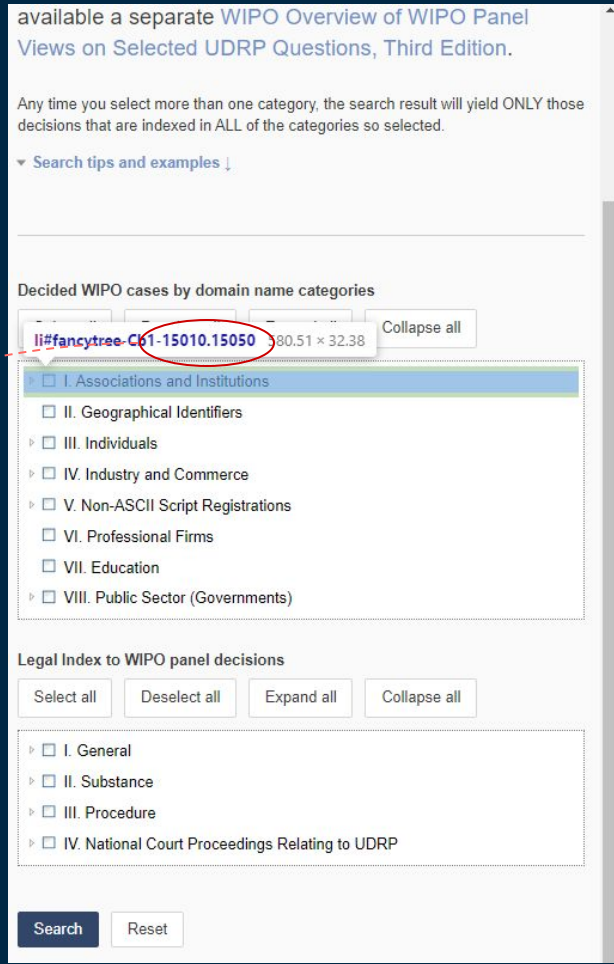
This section of the code allows us to import different packages

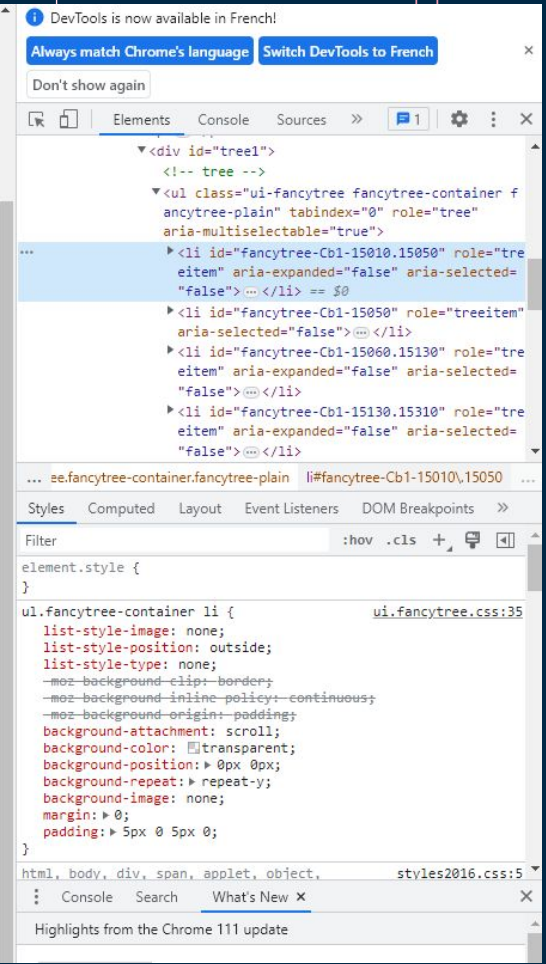
This section of the code opens the web browser, stores the URL in a variable, parses and stores the soup in a variable

This section finds all <li/> attributes in the page.

Why search for “li” and “fancytree”?







SCRAPING : getting the case details for each dispute

```
In [12]: data_rows=[]

In [13]: for link in links:
    ...:     df=pd.read_html(link)
    ...:     driver.get(link)
    ...:     subsoup = BeautifulSoup(driver.page_source)
    ...:     aas = subsoup.find_all("a", string=re.compile("[A-Z]+\d+-\d+"))
    ...:     dd={}
    ...:     for a in aas:
    ...:         href = "https://www.wipo.int/amc/en/domains/search/" + a.get("href")[2:]
    ...:         dd[a.text] = href
    ...:         driver.get(href)
    ...:         case_soup= BeautifulSoup(driver.page_source)
    ...:         table=case_soup.find("table")
    ...:         for tr in table.find_all("tr"):
    ...:             tds=tr.find_all("td")
    ...:             if len(tds)==2:
    ...:                 data_rows.append(tds[1].text.strip())
    ...:

In [14]: len(data_rows)
Out[14]: 22669
```

1

2

3

1

2

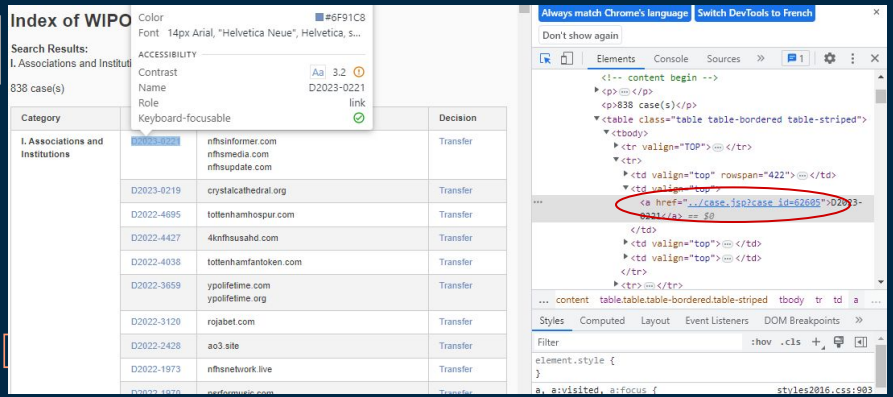
3

This section creates a loop for each link in the list of links

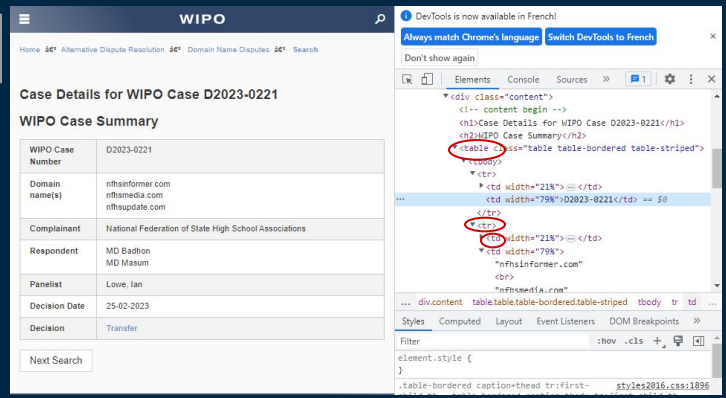
The complement of the link of each case's details can be found in a href

The case details we are interested in are contained in "tr" and then in "td" elements

2



3



SCRAPING : getting the case details for each dispute

```
In [16]: result_list=[]

In [17]: for i in range(12,len(data_rows),7):
    ...:     if isinstance(data_rows[i],str) and len(data_rows[i])==10:
    ...:         continue
    ...:     else:
    ...:         data_rows.insert(i,"NA")
    ...:         result_list.append(i)
    ...:

In [18]: len(data_rows)
Out[18]: 22701
```

Some of the decisions do not have a date (which should be every 7th element on the list data_rows), so we insert “NA” as a placeholder

Data_rows is a list, a continuous sequence of 7 pieces of information regarding 3,243 decisions

```
In [19]: data_rows[0:50]
Out[19]:
['D2023-0221',
 'nfhsinformer.comnfhsmedia.comnfhsupdate.com',
 'National Federation of State High School Associations',
 'MD BadhonMD Masum',
 'Lowe, Ian',
 '25-02-2023',
 'Transfer',
 'D2023-0219',
 'crystalcathedral.org',
 'Crystal Cathedral Ministries',
 'Martina Zammit',
 'Swinson, John',
 '05-03-2023',
 'Transfer',
 'D2022-4695',
 'tottenhamhospur.com',
 'Tottenham Hotspur Limited',
 'Whois Privacy Protection Service by onamae.com / c f, fc',
 'Kondo, Keiji',
 '07-02-2023',
 'Transfer',
 'D2022-4427',
 '4knfhsusahd.com',
 'National Federation of State High School Associations',
 'Erwin Nirwana',
 'K‡kl", Kaya',
 '01-03-2023',
 'Transfer',
 'D2022-4038',
 'tottenhamfantoken.com',
 'Tottenham Hotspur Limited',
 'Tony Fliet',
 'Barbero, Luca',
 '19-12-2022',
 'Transfer',
 'D2022-3659',
 'ypolifetime.comypolifetime.org',
 'YPO, Inc.',
 'Jay SchlimKing Kali',
 'McElwaine, John C',
 '28-11-2022',
 'Transfer',
 'D2022-3120',
 'rojabet.com',
 'Asociacion Nacional de Futbol Profesional',
 'New World Times S.A, New World Times S.A',
 'Campello Estebaranz, Reyes',
 '09-11-2022',
 'Transfer',
 'D2022-2428']
```

SCRAPING : getting the case details for each dispute

```
In [20]: DFS=[data_rows[i:i+7] for i in range(0,len(data_rows),7)]

In [21]: DFS[0:3]
Out[21]:
[['D2023-0221',
  'nfhsinformer.comnfhsmedia.comnfhsupdate.com',
  'National Federation of State High School Associations',
  'MD BadhonMD Masum',
  'Lowe, Ian',
  '25-02-2023',
  'Transfer'],
 ['D2023-0219',
  'crystalcathedral.org',
  'Crystal Cathedral Ministries',
  'Martina Zammit',
  'Swinson, John',
  '05-03-2023',
  'Transfer'],
 ['D2022-4695',
  'tottenhamhospur.com',
  'Tottenham Hotspur Limited',
  'Whois Privacy Protection Service by onamae.com / c f, fc',
  'Kondo, Keiji',
  '07-02-2023',
  'Transfer']]

In [22]: uniqueDFS=[]

In [23]: for sublist in DFS:
    ...:     if sublist not in uniqueDFS:
    ...:         uniqueDFS.append(sublist)
    ...:

In [24]: len(DFS)
Out[24]: 3243

In [25]: len(uniqueDFS)
Out[25]: 2051

In [26]: frameDFS=pd.DataFrame(uniqueDFS, columns=["Case Number", "Domain Name", "Complainant", "Res
pondent","Paneli
    ...: st","Decision Date", "Decision"])

In [27]: frameDFS.to_clipboard(index=False)
```

DFS organizes data_rows in a list of lists, each list containing all the information related to one single decision

uniqueDFS is DFS but without duplicates

Creating a data frame

… LEADING TO OUR FINAL DATAFRAME

	A	B	C	D	E	F	G
1	Case Number	Domain Name	Complainant	Respondent	Panelist	Decision Date	Decision
2	D2023-0221	nfhsinformer.co	National Federa	MD BadhonMD Mas	Lowe, Ian	25-02-2023	Transfer
3	D2023-0219	crystalcathedral.	Crystal Cathedra	Martina Zammit	Swinson, John	5/3/2023	Transfer
4	D2022-4695	tottenhamhospu	Tottenham Hots	Whois Privacy Prote	Kondo, Keiji	7/2/2023	Transfer
5	D2022-4427	4knfhsusahd.con	National Federa	Erwin Nirwana	Köklü, Kaya	1/3/2023	Transfer
6	D2022-4038	tottenhamfantok	Tottenham Hots	Tony Fliet	Barbero, Luca	19-12-2022	Transfer
7	D2022-3659	ypolifetime.com	YPO, Inc.	Jay SchlimKing Kali	McElwaine, John	28-11-2022	Transfer
8	D2022-3120	rojabet.com	Asociacion Naci	New World Times S	Campello Estebai	9/11/2022	Transfer
9	D2022-2428	ao3.site	Organization fo	Domains By Proxy, I	Lowe, Ian	NA	Transfer
10	D2022-1973	nfhsnetwork.live	National Federa	Angga Nuryuana, ng	Kyslyy, Taras	22-08-2022	Transfer
11	D2022-1970	psrformusic.com	PRS for Music Li	Privacy service prov	Smith, Nicholas	14-07-2022	Transfer
12	D2022-1944	goodwilll.org	Goodwill Indust	Federico Murua	Nodine, Lawrenc	3/8/2022	Transfer
13	D2022-1855	rollandgarros.co	Federation Fran	Domain Admin, Wh	Austin, Scott R.	7/7/2022	Transfer
14	D2022-1790	marcet.academy	Fundación Marc	Silvia Martín Martín	Campello Estebai	19-07-2022	Complaint denied
15	D2022-1605	hillsongstor.com	Hillsong Church	杨智超 (Zhi Chao Ya	Simone, Joseph	27-06-2022	Transfer
16	D2022-1314	coopbutikerstorr	Kooperativa För	Ivan Stanojevic	Maier, Steven A.	23-06-2022	Transfer
17	D2022-0653	bacchettafounda	Bacchetta Foun	1&1 Internet Inc. / M	Badgley, Robert A	14-04-2022	Transfer
18	D2022-0390	wwwmychartatr	Rady Children's	蒋黎 (Jiang Li)	Soh, Kar Liang	1/4/2022	Transfer
19	D2022-0348	facareersincoach	The Football As	Domain Admin, Wh	Blackmer, W. Sco	14-03-2022	Transfer
20	D2022-0228	olympic-ioco-ws	International Ol	Ahmed Syed, Adver	Donahey, M. Scot	14-03-2022	Transfer
21	D2022-0209	olympgames.con	The Internation	Privacy service prov	Willoughby, Tony	1/3/2022	Transfer
22	D2021-4176	higherspiritualis	Aquarian Found	Domain Privacy / Ri	Lothian, Andrew	7/3/2022	Complaint denied
23	D2021-3934	fundesarte.org	Fundacion EOI	Kamil Gaede	Bernstein, David	21-12-2021	Transfer
24	D2021-3869	actionlogements	Action Logemer	Jeremie Casalle	Nappey, Alexand	15-02-2022	Transfer
25	D2021-3619	liverpoolfc.footb	The Liverpool F	Amre Salim	Lowe, Ian	30-12-2021	Transfer
26	D2021-3598	aoga.com	Alaska Oil and G	Obada Alzatari	Perraki, Marina	7/2/2022	Transfer
27	D2021-3529	eastwestchurch.(	East-West Wors	Withheld for Privac	Badgley, Robert A	28-12-2021	Transfer
28	D2021-3024	gildasclub.org	Cancer Support	Jacob Kovacs	Swinson, John	4/11/2021	Transfer
29	D2021-2784	gia-jewelry.com	Gemological Ins	陈家伟 (Chen Jia W	Wong Wai Man, D	1/11/2021	Transfer
30	D2021-1742	moanapacific.org	Association of A	Robert Allman, Moa	Blackmer, W. Sco	7/9/2021	Complaint denied
31	D2021-1597	psg-france-acade	Paris Saint-Gern	Pascal Lussiana	Vivant, Michel	2/7/2021	Transfer
32	D2021-1392	farmlandreserve	Farmland Reser	Privacy Administrat	Swinson, John	26-06-2021	Transfer
33	D2021-1321	parissaintgermai	Paris Saint-Gern	Contact Privacy Inc.	Jabur, Wilson Pin	9/6/2021	Transfer
34	D2021-1115	imaocolympic.co	The Internation	Domain Admin, FBS	Köklü, Kaya	8/6/2021	Transfer

Total of 2,052 lines

len(uniqueDFS) + title of column

Our Analysis

Thanks to Statistics

03

Yearly evolution of the number of decisions

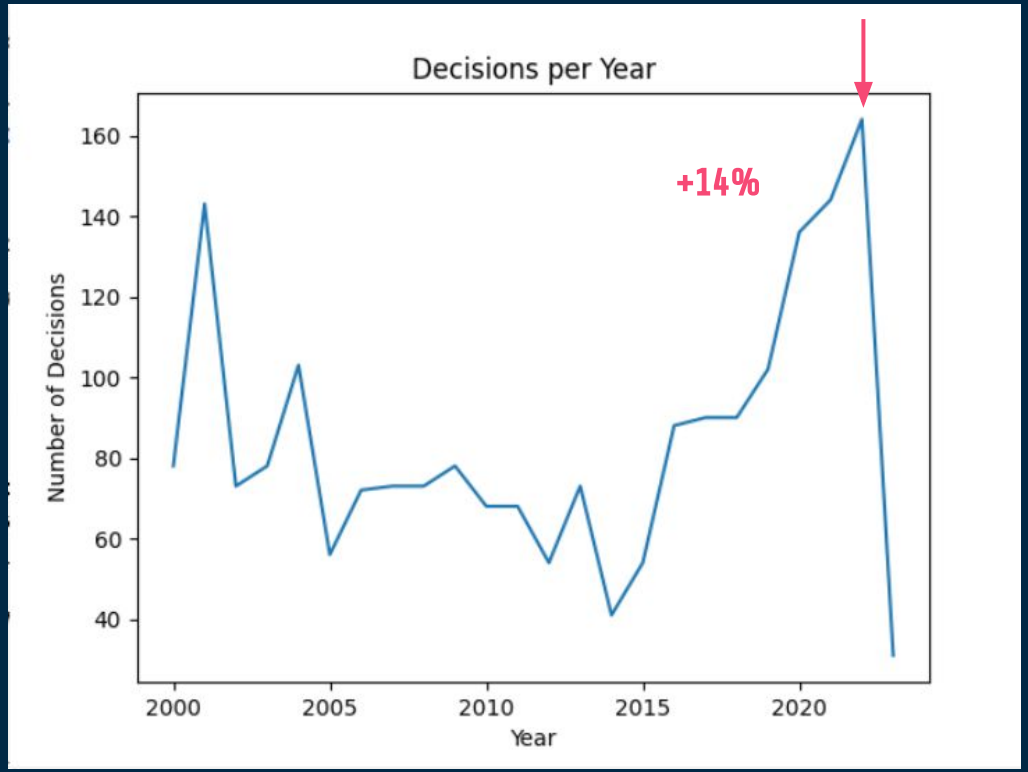
```
#Displaying a chart that shows the evolution of the number of decisions throughout
the years

frameDFS[“Decision Date”]=pd.to_datetime(frameDFS[“Decision Date”], errors=
“coerce”)
mask= frameDFS[“Decision Date”].notna()
df=frameDFS.loc[mask, [“Decision Date”]]
df[“Decision Date”]=pd.to_datetime(df[“Decision Date”])
fig, ax= plt.subplots()
ax.plot(decisions_per_year.index, decisions_per_year.values)
ax.set_xlabel(“Year”)
ax.set_ylabel(“Number of Decisions”)
ax.set_title(“Decisions per Year”)
plt.show()
```

We first need to mask the “NA” elements where there was no date

“df” is a data frame with the decision dates

We use matplotlib.pyplot to create a graph

A surge of decisions in the IP world in 2020 & 2021



Main reason:
cybersquatting
related to Covid-19

COVID-19 malicious domain names classification

Paul K. Mvula, Paula Branco, Guy-Vincent Jourdan, Herna L. Viktor

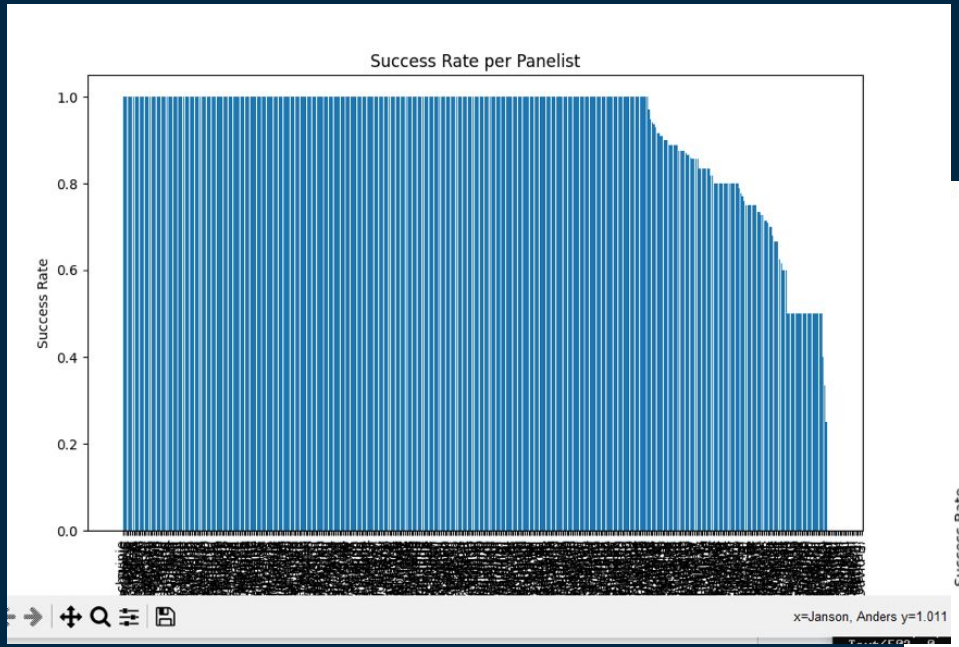
School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON K1N 6N5, Canada

Received 22 April 2021, Revised 4 May 2022, Accepted 7 May 2022, Available online 20 May 2022, Version of Record 30 May 2022.

Success rate by panelist

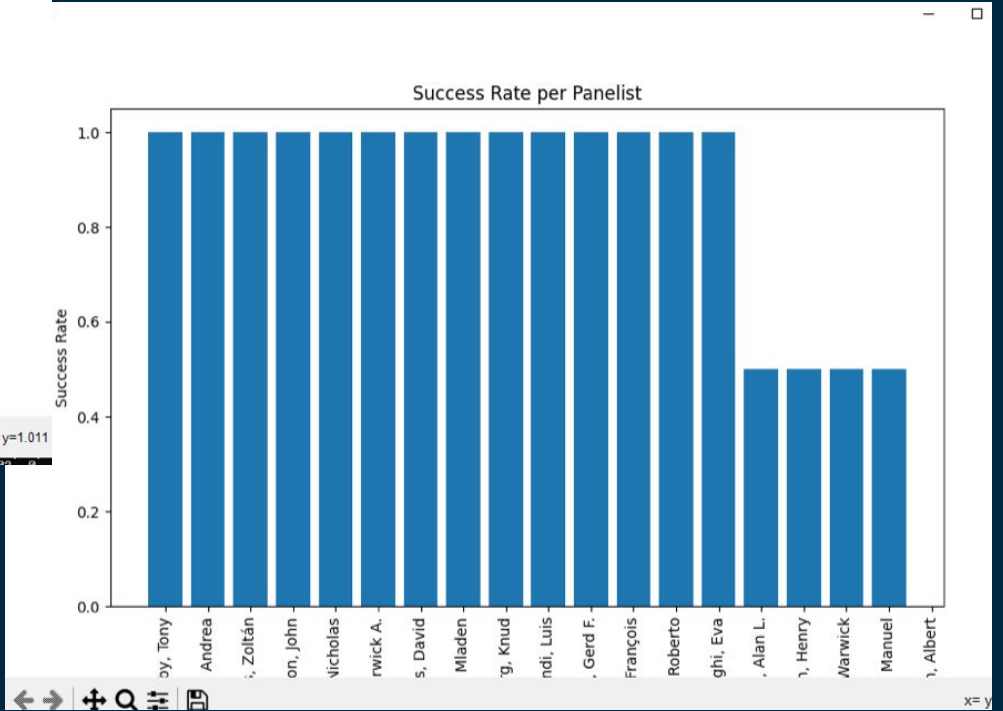
```
# displaying the success rate of decisions per panelist

# Count the number of decisions and favorable decisions per panelist
decisions_per_panelist = frameDFS['Panelist'].value_counts()
favorable_decisions_per_panelist = frameDFS[frameDFS['Decision'] ==
'Transfer']['Panelist'].value_counts()

# Calculate the success rate per panelist
success_rate_per_panelist = favorable_decisions_per_panelist / decisions_per_panelist

# Sort the panelists by success rate
success_rate_per_panelist = success_rate_per_panelist.sort_values(ascending=False)

# Create a bar plot of the success rate per panelist
plt.figure(figsize=(10, 6))
plt.bar(success_rate_per_panelist.index, success_rate_per_panelist.values)
plt.title('Success Rate per Panelist')
plt.xlabel('Panelist')
plt.ylabel('Success Rate')
plt.xticks(rotation=90)
plt.show()
```

Success rate per panelist: a narrower focus per category



All categories:
not readable

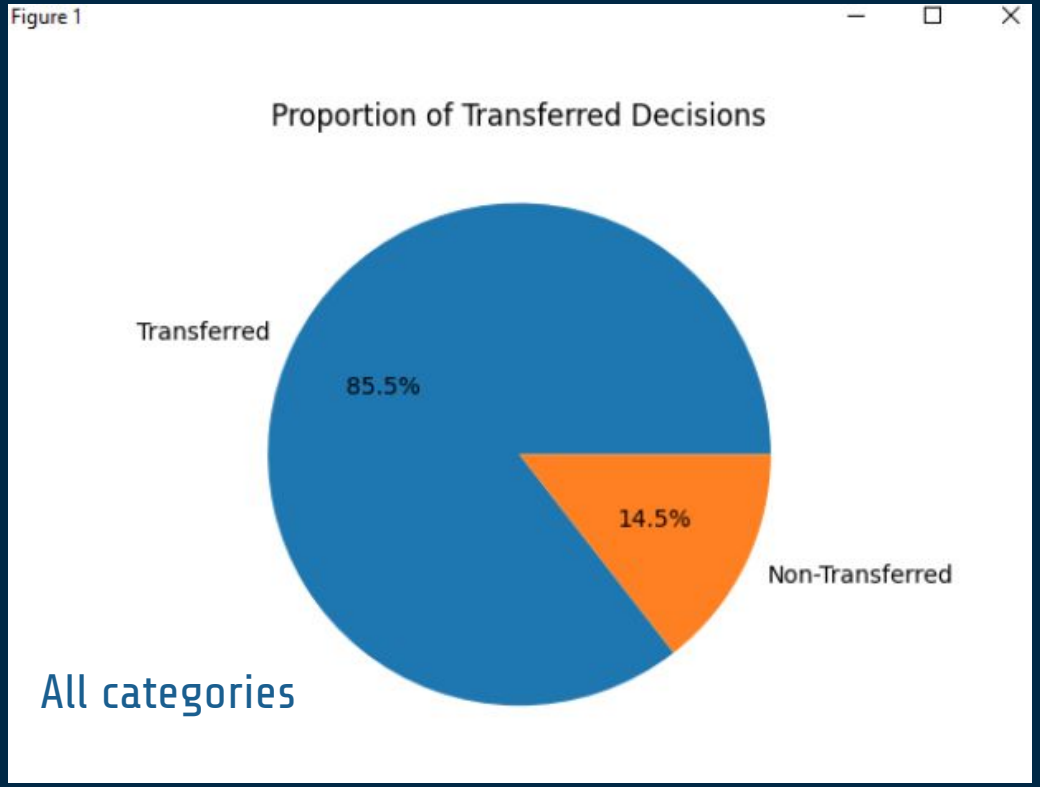
Geographical identifiers category



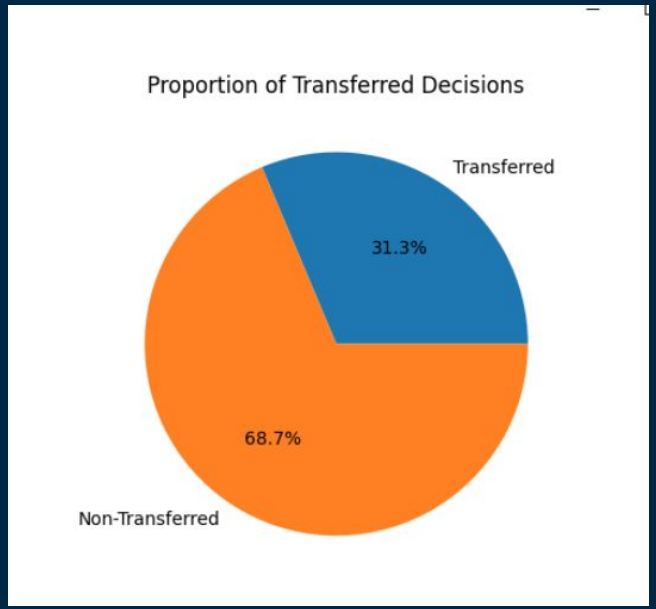
Transferred decisions

```
#creating a pie chart showing the proportion of transferred decisions compared to the total number of decisions made

# Counting the number of transferred decisions
transferred_count = sum(frameDFS['Decision'] == 'Transfer')

# Calculating the proportion of transferred decisions
total_count = len(frameDFS)
transferred_proportion = transferred_count / total_count

# Calculating the proportion of non-transferred decisions
non_transferred_proportion = 1 - transferred_proportion

# Creating labels for the pie chart
labels = ['Transferred', 'Non-Transferred']

# Creating values for the pie chart
values = [transferred_proportion, non_transferred_proportion]

# Creating the pie chart
plt.pie(values, labels=labels, autopct='%1.1f%%')
plt.title('Proportion of Transferred Decisions')

# Displaying the chart
plt.show()
```

A huge proportion of decisions being transferred

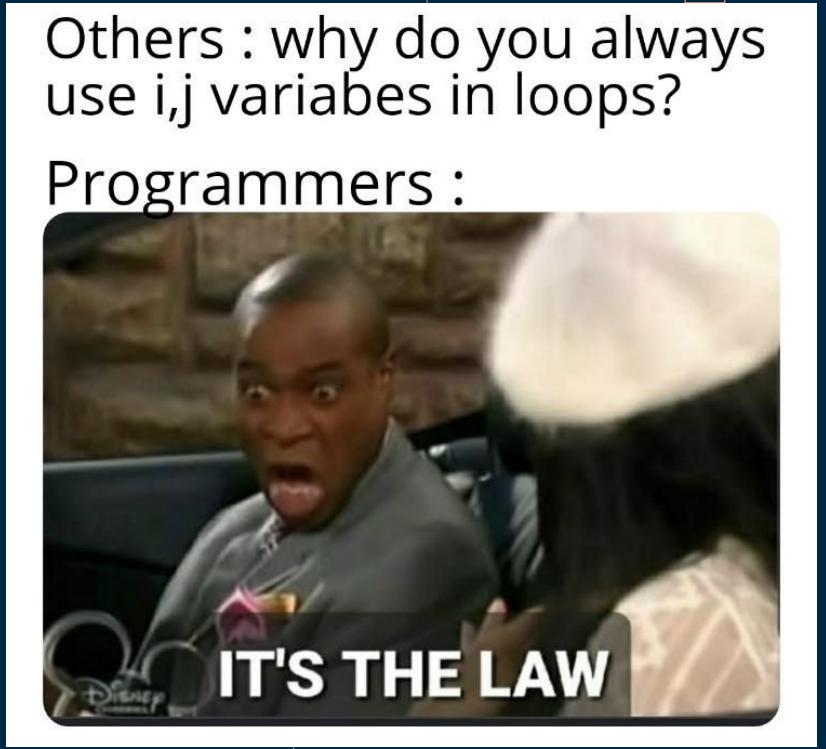


All categories



Geographical identifiers category
Domain names too generic

The Mighty Pythons thank you for your attention !



slidesgo