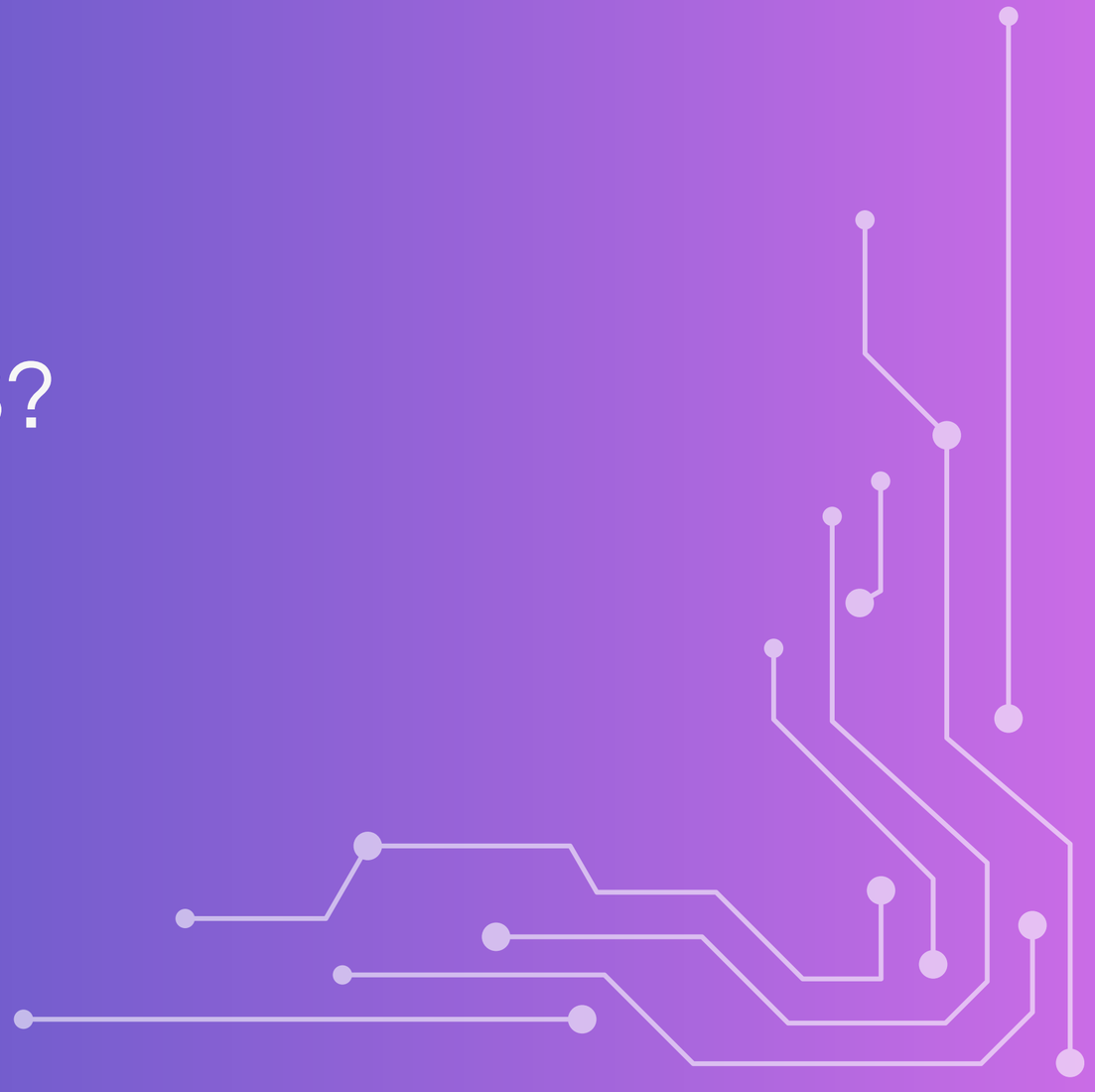
MARCH 26, 2024

THE ABUSE OF LAW PROCEDURE: A DOUBLE EDGED-SWORD FOR THE FRENCH TAX AUTHORITIES?

JOANNE CHUN MASGNAUX

ELISE CRIEZ



Agenda

0.

WHAT IS OUR PROJECT ABOUT?

Our project in a nutshell

Purpose: analyze, on the basis of court decisions, the success rate of the French Tax Authorities (FTA) when adjusting a taxpayer on the grounds of the abuse of law procedure.

The abuse of law procedure is defined in **article L.64** of the French Tax Procedures Code.

This procedure empowers the tax **authorities to:**

(i) **Disregard** actions undertaken by a taxpayer **solely for tax purposes**;

(ii) **Rectify the taxpayer's position** (often accompanied by substantial penalties).

Example: To avoid an increase in their real estate wealth tax (IFI), a taxpayer may have someone with lesser wealth purchase a building and pay the full price to them.

Of particular complexity for the tax authorities is demonstrating the taxpayer's sole tax motive.

1. IMPORTING LIBRARIES

Importing Libraries

We begin by importing the necessary libraries which provides classes and methods for different tasks.

Basically, we set up an environment for web scraping, data analysis, visualization, and statistical analysis.

```
from selenium import webdriver
from selenium.webdriver.support.ui import Select
from selenium.common.exceptions import NoSuchElementException
from selenium.webdriver.common.by import By

from bs4 import BeautifulSoup
import pandas as pd
import time
import os

import regex as re
import requests

import seaborn as sns
import matplotlib.pyplot as plt

from scipy.stats import chi2_contingency
from datetime import datetime

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC, SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
import spacy
from collections import Counter

from spacy.lang.fr.stop_words import STOP_WORDS as fr_stop
import numpy as np

pd.set_option('display.max_rows', 300)
pd.set_option('display.max_colwidth', 300)
```

2. SCRAPING DATA FROM THE WEB

Scraping data

The code sets up a connection to the Ariane website, configures some options for the Chrome Browser, opens the website, and performs a search for "L.64".

01

We use a special app called "Web driver" to interact with the Chrome browser.

02

We customize some options for the Chrome browser using `chromeOptions`.

03

We open Ariane (`driver.get(website)`).

```
#Website and path to chrome driver#
website = 'https://www.conseil-etat.fr/arianeweb/#/recherche'
path = '/Users/el/Downloads/chromedriver-mac-arm64/chromedriver' #To be modified with the path of chromedriver
```

```
#Options#

chromeOptions = webdriver.ChromeOptions()
prefs = {"download.default_directory" : "/Users/el/Library/CloudStorage/OneDrive-Personal/Project Text Mining/html"} #Download folder#
chromeOptions.add_experimental_option("prefs",prefs)
driver = webdriver.Chrome(path,options=chromeOptions)
driver.get(website)
```

Scraping data

The code sets up a connection to the Ariane website, configures some options for the Chrome Browser, opens the website, and performs a search for "L.64".

04

We locate the search bar on the Ariane website using its HTML path, specifically targeting the text area where users input their search queries.

- We clear any existing text in the search bar.

- Then we input our search term, in this case, "L.64".

```
#Search bar (to put the "L.64")
textarea = driver.find_element(By.XPATH, "//textarea[@ng-model='search.currentSearch.query']")
textarea.clear()  # Clear any pre-filled text
textarea.send_keys('"L.64"')  # Key word to search
```

"L.64"

Correction orthographique Pluriels, féminins, conjugaisons Acronymes +

Fonds de jurisprudence

- [x] Décisions du Conseil d'État ?
- [ ] Analyses du Conseil d'État ?
- [ ] Conclusions des rapporteurs publics ?

- [ ] Décisions du Tribunal des conflits ?
- [ ] Analyses du Tribunal des conflits ?

Les décisions et analyses de 1821 à 1954 > Gallica

Options de recherche

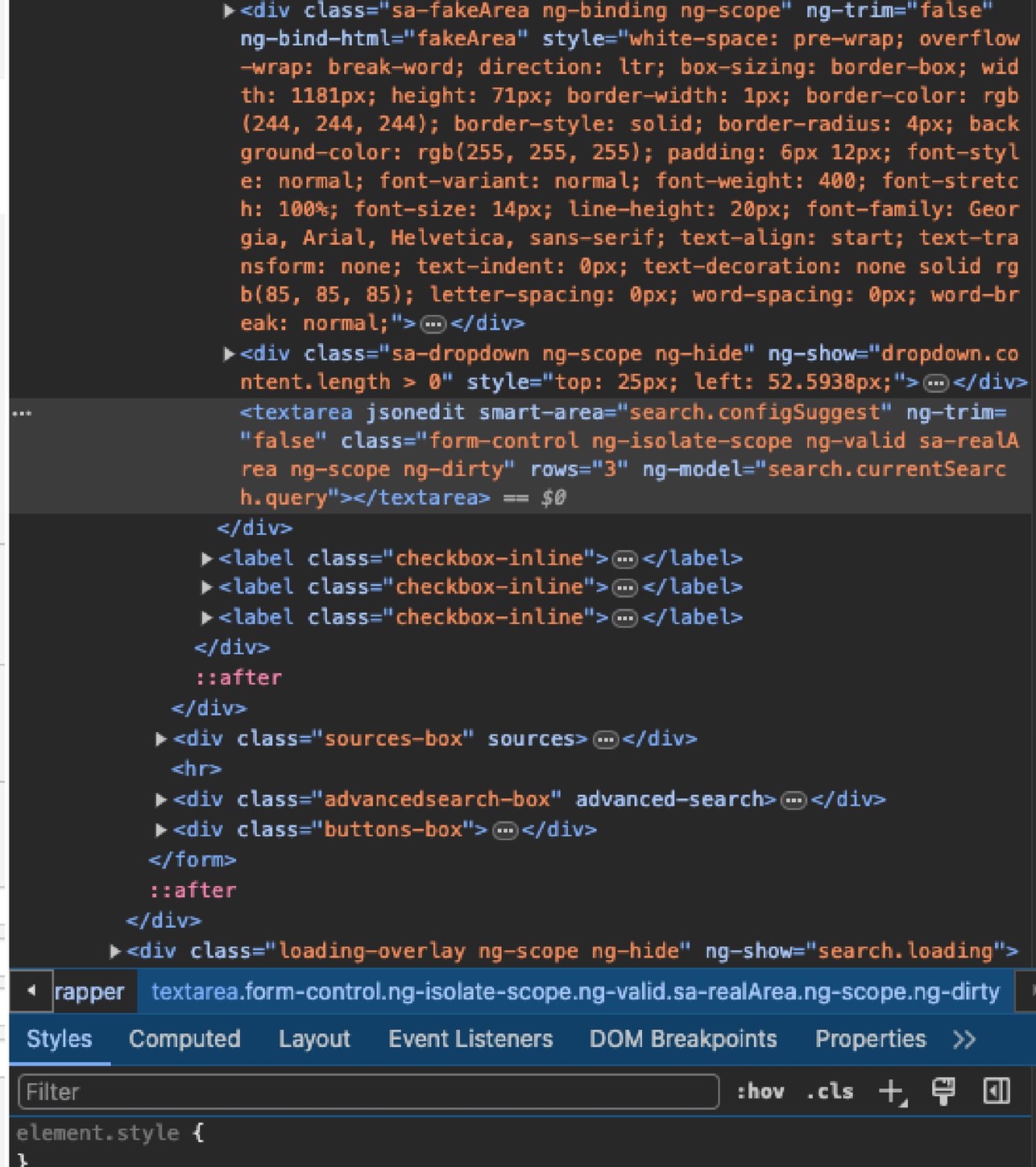
Numéro d'affaire

Formation de jugement

Abstract

Juridiction

Plan de classement de la juridiction administrative



Scraping data

Here, our code automates the process of entering a search term, selecting checkboxes, and clicking the search button on a webpage, all using Python and Selenium.

05

We inspect the HTML code of the webpage to identify the attributes and values for the "***Décisions du Conseil d'Etat***" and "***Arrêts de CAA***" checkboxes.

06

We ask Python to find the checkbox elements based on these values.

07

We ask Python to click on the checkboxes.

```
# looking at the .html in a browser > Inspect, we see the relevant checkbow has an
# attribute 'ng-change' with value 'sources.selectSource('AW_DCE')'
att = "sources.selectSource('AW_DCE')"
att2 = "sources.selectSource('AW_DCA')"

el = driver.find_element(By.XPATH, r'.//*[@ng-change="' + att + '"]')
el.click()

el2 = driver.find_element(By.XPATH, r'.//*[@ng-change="' + att2 + '"]')
el2.click()
```

CONSEIL D'ÉTAT

Ariane web

"L.64"

Correction orthographique Pluriels, féminins, conjugaisons Acronymes +

input.ng-valid.ng-dirty 13 × 13

Décisions du Conseil d'État ?
Analyses du Conseil d'État ?
Conclusions des rapporteurs publics ?

Arrêts des cours administratives d'appel ?
Analyses des cours administratives d'appel ?

Décisions du Tribunal des conflits ?
Analyses du Tribunal des conflits ?

Les décisions et analyses de 1821 à 1954 > Gallica

Options de recherche

Numéro d'affaire ?
Formation de jugement ?
Abstract ?
Juridiction ?
Plan de classement de la juridiction administrative

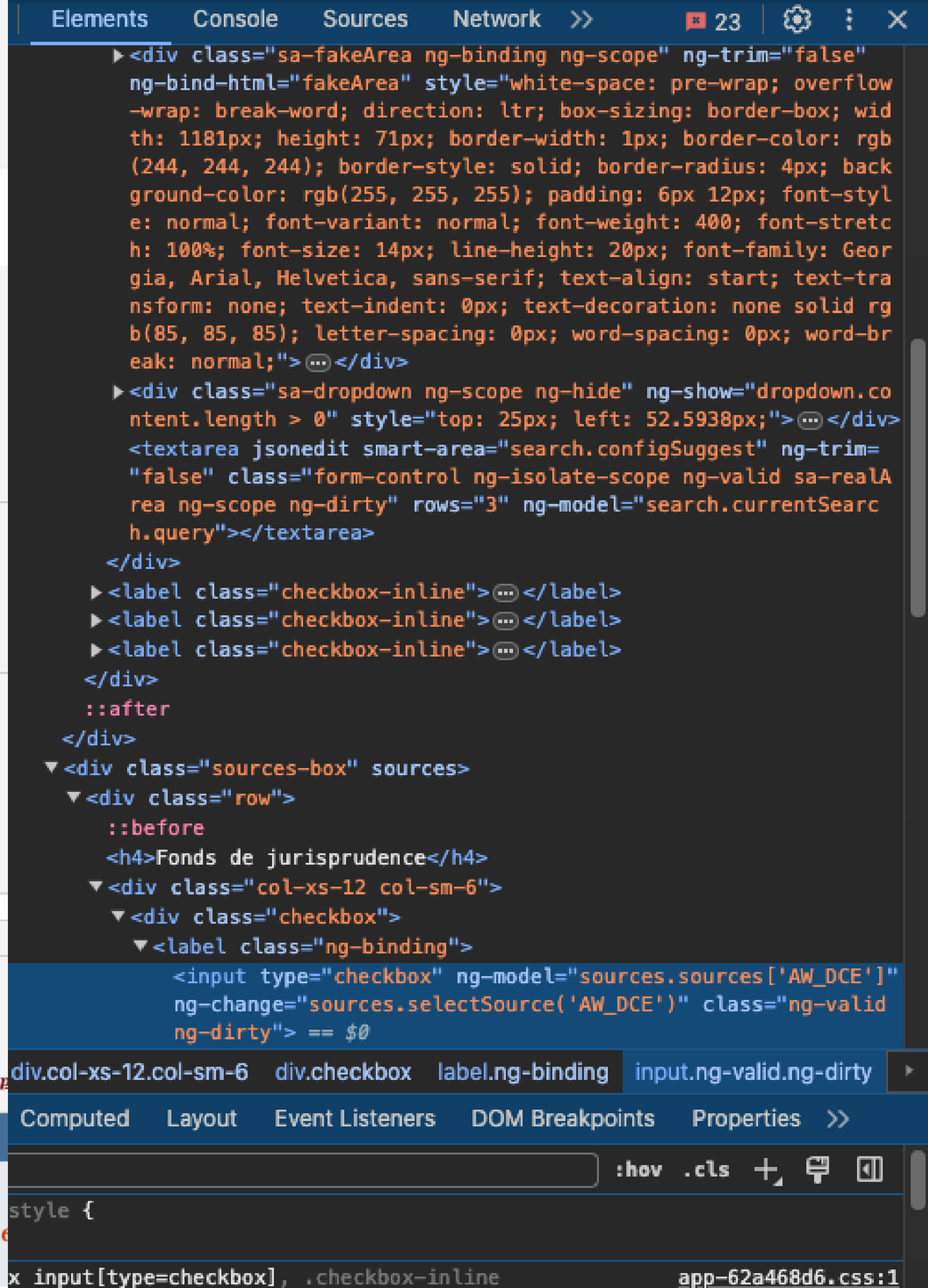
Date de lecture entre le
Code de publication
Sens de la décision

439 résultats

> Imp

	Fonds	Juridiction	Formation de jugement	Date de lecture	Numéro d'affaire
1	Arrêts	CAA NANCY	Chambre	21/12/2023	21NC02477

l'administration a mis ... prévues par l'article L. 64 du livre ... procédures fiscales ... En ce qui ... tiré de l'article L. 64 du livre ... fiscales ... Aux termes de l'article L. 6
résulte également ... précitées de l'article L. 64 du livre ... fiscales.



CONSEIL D'ÉTAT

Ariane web

"L.64"

Correction orthographique Pluriels, féminins, conjugaisons Acronymes +

Fonds de jurisprudence

- Décisions du Conseil d'État ?
- Analyses du Conseil d'État ?
- Conclusions des rapporteurs publics ?
- Arrêts des cours administratives d'appel ?
- Analyses des cours administratives d'appel ?
- Décisions du Tribunal des conflits ?
- Analyses du Tribunal des conflits ?

Les décisions et analyses de 1821 à 1954 > Gallica

Options de recherche

Numéro d'affaire ?
Formation de jugement ?
Abstract ?
Juridiction ?
Plan de classement de la juridiction administrative
Date de lecture entre le
Code de publication
Sens de la décision

439 résultats

> Im

	Fonds	Juridiction	Formation de jugement	Date de lecture	Numéro d'affaire
1	Arrêts	CAA NANCY	Chambre	21/12/2023	21NC02477

l'administration a mis ... prévues par l'article L. 64 du livre ... procédures fiscales ... En ce qui ... tiré de l'article L. 64 du livre ... fiscales ... Aux termes de l'article L. résulte également ... précitées de l'article L. 64 du livre ... fiscales.

```
<div class="gallica-link">…</div>
</div>
<div class="col-xs-12 col-sm-6">
<div class="checkbox">
<label class="ng-binding">
<input type="checkbox" ng-model="sources.sources['AW_DCA']"
ng-change="sources.selectSource('AW_DCA')" class="ng-valid
ng-dirty"> == $0
" Arrêts des cours administratives d'appel"
</label>
<span class="info" ng-click="sources.info('AW_DCA')">?</span>
</div>
<div class="checkbox">…</div>
<div class="checkbox"></div>
<hr>
</div>
::after
</div>
</div>
<hr>
<div class="advancedsearch-box" advanced-search>…</div>
<div class="buttons-box">…</div>
</form>
```

div.col-xs-12.col-sm-6 div.checkbox label.ng-binding input.ng-valid.ng-dirty

Computed Layout Event Listeners DOM Breakpoints Properties >>

:hov .cls

```
style {

x input[type=checkbox], .checkbox-inline                app-62a468d6.css:1
ype=checkbox] {
cal-align: sub;
ion: static;

x input[type=checkbox], .checkbox-inline                vendor-d6fcedf8.css:15
ype=checkbox], .radio input[type=radio], .radio-
nput[type=radio] {
ion: absolute;
n-left: -20px;
n-top: 4px \9;
```

Scraping data

Here, our code automates the process of browsing through cases on the webpage, clicking through pages, and performing actions on each page, all using Python and Selenium.

08

The code uses BeautifulSoup to parse and extract the HTML content of the webpage.

09

The table data is converted into a pandas dataframe.

```
#Open and download all cases into the download folder#
#Open the first case and click next to surf through all cases#


soup = BeautifulSoup(driver.page_source)  # We recreate a soup, now that the page source
table = soup.find_all("table")[-1]  # Collect tables from the page; there are two of them
# in the page source (and find_all returns a list), and we are interested in the last one.

df = pd.read_html(str(table))[-1]
# Convert the table in a panda
# dataframe, with each row storing data about one decision.
```

Scraping data

10

For each row in the dataframe (representing a case), Python:

- Extracts the case number;
- Locates and clicks on the element corresponding to the case number, which loads the page with the judgment.

```python
for index, row in df.iloc[:1].iterrows():
    # For each row, we'll make the browser click on the element and collect the judgment
    num = re.search(r"\d+", row["Numéro d'affaire"]).group()
    row_el = driver.find_element(By.XPATH, ".//td[contains(text(), '" + num + "')]")
    # With that num, we look for the relevant element in browser
    row_el.click()  # load page with judgment
    # time.sleep(1)
    driver.switch_to.window(driver.window_handles[-1])
    ## Switch the driver's focus to the window you just opened, with method "switch to",
    # and argument the relevant window from the list of window_handles
    # (latest loaded window will be -1)

while True:
    # Attempt to find the "next page" button
    try:
        next_page = driver.find_element(By.CSS_SELECTOR, "button[title='document suivant']")
        if next_page.get_attribute('disabled') == 'true' or not next_page.is_displayed():
            # If the "next page" button is disabled or not displayed, break the loop
            break
        else:
            # If the button is active, click it to go to the next page
            next_page.click()
            time.sleep(1)  # Adjust time as necessary for page loading

        # Your logic here for processing the current page
        ave_el = driver.find_element(By.CSS_SELECTOR, "button[title='enregistre le document']")
        ave_el.click()
        # Add any other actions you need to perform on each page

    except NoSuchElementException:
        # If the "next page" button cannot be found at all, assume end of pages
        break
```

Scraping data

11

Python navigates through each case (page), by clicking the “next page” button.

It performs any necessary actions on each page, such as saving the document.

```
for index, row in df.iloc[:1].iterrows():
    # For each row, we'll make the browser click on the element and collect the judgment
    num = re.search(r"\d+", row["Numéro d'affaire"]).group()
    row_el = driver.find_element(By.XPATH, ".//td[contains(text(), '" + num + "')]")
    # With that num, we look for the relevant element in browser
    row_el.click()  # load page with judgment
    # time.sleep(1)
    driver.switch_to.window(driver.window_handles[-1])
    ## Switch the driver's focus to the window you just opened, with method "switch to",
    # and argument the relevant window from the list of window_handles
    # (latest loaded window will be -1)

while True:
    # Attempt to find the "next page" button
    try:
        next_page = driver.find_element(By.CSS_SELECTOR, "button[title='document suivant']")
        if next_page.get_attribute('disabled') == 'true' or not next_page.is_displayed():
            # If the "next page" button is disabled or not displayed, break the loop
            break
        else:
            # If the button is active, click it to go to the next page
            next_page.click()
            time.sleep(1)  # Adjust time as necessary for page loading

        # Your logic here for processing the current page
        ave_el = driver.find_element(By.CSS_SELECTOR, "button[title='enregistre le document']")
        ave_el.click()
        # Add any other actions you need to perform on each page

    except NoSuchElementException:
        # If the "next page" button cannot be found at all, assume end of pages
        break
```

Conseil d'État

N° 471003
ECLI:FR:CECHR:2023:471003.20230929
Mentionné aux tables du recueil Lebon

8ème - 3ème chambres réunies

M. Rémy Schwartz, président
Mme Alianore Descours, rapporteur
Mme Karin Ciavaldini, rapporteur public
SCP CELICE, TEXIDOR, PERIER, avocats

Lecture du vendredi 29 septembre 2023

REPUBLIQUE FRANCAISE

AU NOM DU PEUPLE FRANCAIS

Vu la procédure suivante :

M. A... B... et Mme C... D... ont demandé au tribunal administratif de Lyon de prononcer la décharge des cotisations supplémentaires d'impôt sur le revenu et de prélèvements sociaux auxquelles ils ont été assujettis au titre de l'année 2015, ainsi que des pénalités correspondantes. Par un jugement n° 2001629 du 2 mars 2021, ce tribunal a rejeté leur demande.

Par un arrêt n° 21LY01365 du 15 décembre 2022, la cour administrative d'appel de Lyon, sur appel de M. B... et Mme D..., après avoir prononcé un non-lieu à statuer à concurrence d'un dégrèvement intervenu en cours d'instance, leur a accordé la décharge du surplus des cotisations supplémentaires d'impôt sur le revenu et de contributions sociales auxquelles ils ont été assujettis au titre de l'année 2015, ainsi que des pénalités correspondantes.

Par un pourvoi et un mémoire en réplique, enregistrés les 2 février et 31 juillet 2023 au secrétariat du contentieux du Conseil d'Etat, le ministre de l'économie, des finances et de la souveraineté industrielle et numérique demande au Conseil d'Etat :

3. CREATING A DATA FRAME

Creating a data frame

This code block parses the HTML files we just downloaded:
It defines regex patterns and, using these patterns, extracts relevant information from the HTML files to create a DataFrame.

```
# Create dictionnary of months in french to be converted in number
months_fr_to_en = {
    'janvier': '01', 'février': '02', 'mars': '03', 'avril': '04',
    'mai': '05', 'juin': '06', 'juillet': '07', 'août': '08',
    'septembre': '09', 'octobre': '10', 'novembre': '11', 'décembre': '12'
}

# Regex pattern to match French date format "Lecture du DD Month YYYY", ignoring case and day of the week
date_pattern = re.compile(r'Lecture du\s+(?:\w+\s+)?(\d+)\s+(\w+)\s+(\d{4})', re.IGNORECASE)

# Path to the folder containing your HTML files
directory_path = '/Users/el/Library/CloudStorage/OneDrive-Personal/Project Text Mining/html'

# Initialize a list to hold the data
data = []

# Regex patterns to identify the participant, the body, and the decision text starting with "Article 1 :"
participant_pattern = re.compile(r'(.*?)REPUBLIQUE FRANCAISE.*?AU NOM DU PEUPLE FRANCAIS', re.DOTALL | re.IGNORECASE)
body_pattern = re.compile(r'REPUBLIQUE FRANCAISE.*?AU NOM DU PEUPLE FRANCAIS(.*?)(?=Article 1(?:er)?[\s:,.])', re.DOTALL | re.IGNORECASE)
decision_pattern = re.compile(r'(Article 1(?:er)?[\s:,.].*)', re.DOTALL | re.IGNORECASE)
```

Creating a data frame

Voir aussi...

Conseil d'État

N° 471003
ECLI:FR:CECHR:2023:471003.20230929
Mentionné aux tables du recueil Lebon

M. Rémy Schwartz, président
Mme Alianore Descours, rapporteur
Mme Karin Ciavaldini, rapporteur public
SCP CELICE, TEXIDOR, PERIER, avocats

Lecture du vendredi 29 septembre 2023

participant_pattern

REPUBLIQUE FRANCAISE

AU NOM DU PEUPLE FRANCAIS

Vu la procédure suivante :

M. A... B... et Mme C... D... ont demandé au tribunal administratif de Lyon de prononcer la décharge des cotisations supplémentaires d'impôt sur le revenu et de prélèvements sociaux auxquelles ils ont été assujettis au titre de l'année 2015, ainsi que des pénalités correspondantes. Par un jugement n° 2001629 du 2 mars 2021, ce tribunal a rejeté leur demande.

Par un arrêt n° 21LY01365 du 15 décembre 2022, la cour administrative d'appel de Lyon, sur appel de M. B... et Mme D..., après avoir prononcé un non-lieu à statuer à concurrence d'un dégrèvement intervenu en cours d'instance, leur a accordé la décharge du surplus des cotisations supplémentaires d'impôt sur le revenu et de contributions sociales auxquelles ils ont été assujettis au titre de l'année 2015, ainsi que des pénalités correspondantes.

Par un pourvoi et un mémoire en réplique, enregistrés les 2 février et 31 juillet 2023 au secrétariat du contentieux du Conseil d'Etat, le ministre de l'économie, des finances et de la souveraineté industrielle et numérique demande au Conseil d'Etat :

1°) d'annuler les articles 2 et 3 de cet arrêt ;

2°) réglant l'affaire au fond dans cette mesure, de rejeter l'appel de M. B... et Mme D....

body_pattern

decision_pattern

D E C I D E :

Article 1er : Le pourvoi du ministre de l'économie, des finances et de la souveraineté industrielle et numérique est rejeté.
Article 2 : L'Etat versera à M. B... et Mme D... une somme de 3 000 euros au titre de l'article L. 761-1 du code de justice administrative.
Article 3 : La présente décision sera notifiée au ministre de l'économie, des finances et de la souveraineté industrielle et numérique et à M. A... B... et Mme C... D....
Délibéré à l'issue de la séance du 18 septembre 2023 où siégeaient : M. Rémy Schwartz, président adjoint de la section du contentieux, présidant ; M. Pierre Collin, M. Stéphane Verclytte, présidents de chambre ; M. Jonathan Bosredon, M. Hervé Cassagnabère, M. Christian Fournier, M. Frédéric Gueudar Delahaye, Mme Françoise Tomé, conseillers d'Etat et Mme Alianore Descours, maître des requêtes en service extraordinaire-rapporteure.

Creating a data frame

The code defines a function (`determine_outcome`) to determine the outcome of each decision based on the decision text.

The function splits the decision text into sentences and checks for keywords like "*rejeté*" (rejected) or "*annulé*" (cancelled) to determine whether the outcome favors the "*Contribuable*" (taxpayer) or the "*Administration* "(tax authorities).

```python
# Function to determine the outcome with refined logic
def determine_outcome(decision_text):
    # Split the decision text into sentences for a detailed analysis.
    sentences = re.split(r'(?<=[.!?])\s+', decision_text)

    for sentence in sentences:
        # Check for "rejeté" or "rejetée" in the sentence.
        if re.search(r'rejeté|rejetée', sentence, re.IGNORECASE):
            # If "ministre" is in the same sentence, outcome is "Contribuable".
            if 'ministre' in sentence.lower():
                return "Contribuable"
            # If "ministre" is not mentioned, outcome is "Administration".
            else:
                return "Administration"
        # Check for other keywords indicating a "Contribuable" outcome.
        elif re.search(r'annulé|annulée|admises?|admis', sentence, re.IGNORECASE):
            return "Contribuable"

    # If none of the conditions are met, the outcome is "Indéterminé".
    return "Indéterminé"
```

Creating a data frame

Based on the previously defined patterns, this code block extracts relevant information from the HTML files to create a DataFrame.

```
# Iterate over each file in the directory
for filename in os.listdir(directory_path):
    if filename.endswith('.html'):
        file_path = os.path.join(directory_path, filename)

        with open(file_path, 'r', encoding='ISO-8859-1') as file:
            soup = BeautifulSoup(file.read(), 'html.parser')
            text_content = soup.get_text(" ", strip=True)

            participant_text = participant_pattern.search(text_content).group(1).strip() if participant_pattern.search(text_content) else "Not
            body_text = body_pattern.search(text_content).group(1).strip() if body_pattern.search(text_content) else "Not found"
            decision_text = decision_pattern.search(text_content).group(1).strip() if decision_pattern.search(text_content) else "Not found"

            date_match = date_pattern.search(participant_text)
            formatted_date = "Not found"
            if date_match:
                day, month_text, year = date_match.groups()
                month = months_fr_to_en.get(month_text.lower(), '00')
                formatted_date = f"{day}/{month}/{year}"

            case_type = "CAA" if any(c.isalpha() for c in os.path.splitext(filename)[0]) else "Decision"
            outcome = determine_outcome(decision_text)

            data.append([filename.split('.')[0], formatted_date, case_type, participant_text, body_text, decision_text, outcome])
```

Creating a data frame

It appends the extracted data for each file to a list called `data`.

It creates a DataFrame (`df`) from the collected data with appropriate column names.

```
# Iterate over each file in the directory
for filename in os.listdir(directory_path):
    if filename.endswith('.html'):
        file_path = os.path.join(directory_path, filename)

        with open(file_path, 'r', encoding='ISO-8859-1') as file:
            soup = BeautifulSoup(file.read(), 'html.parser')
            text_content = soup.get_text(" ", strip=True)

            participant_text = participant_pattern.search(text_content).group(1).strip() if participant_pattern.search(text_content) else "Not
            body_text = body_pattern.search(text_content).group(1).strip() if body_pattern.search(text_content) else "Not found"
            decision_text = decision_pattern.search(text_content).group(1).strip() if decision_pattern.search(text_content) else "Not found"

            date_match = date_pattern.search(participant_text)
            formatted_date = "Not found"
            if date_match:
                day, month_text, year = date_match.groups()
                month = months_fr_to_en.get(month_text.lower(), '00')
                formatted_date = f"{day}/{month}/{year}"

            case_type = "CAA" if any(c.isalpha() for c in os.path.splitext(filename)[0]) else "Decision"
            outcome = determine_outcome(decision_text)

            data.append([filename.split('.')[0], formatted_date, case_type, participant_text, body_text, decision_text, outcome])
```

Creating a data frame

It appends the extracted data for each file to a list called `data`.

It creates a DataFrame (`df`) from the collected data with appropriate column names.

	Case Number	Date	Case	Participant Text	Body Text	Decision Text	Outcome	Avocat
0	313139	8/10/2010	Decision	Conseil d'État N° 313139 ECLI:FR:CESSR:2010:313139.20101008 Mentionné au tables du recueil Lebon 8ème et 3ème sous-sections réunies M. Arrighi de Casanova, président M. Patrick Quinqueton, rapporteur M. Olléon Laurent, rapporteur public SCP ORTSCHEIDT, avocats Lecture du vendredi 8 octobre 2010	Vu le pourvoi, enregistré le 8 février 2008 au secrétariat du contentieux du Conseil d'Etat, présenté par le MINISTRE DU BUDGET, DES COMPTES PUBLICS ET DE LA FONCTION PUBLIQUE ; le ministre demande au Conseil d'Etat d'annuler l'arrêt du 11 décembre 2007 par lequel, faisant droit à la requête de ...	Article 1er : L'arrêt de la cour administrative d'appel de Douai du 11 décembre 2007 est annulé. Article 2 : M. et Mme Bauchart sont déchargés des cotisations supplémentaires d'impôt sur le revenu et de contributions sociales auxquelles ils sont restés assujettis au titre de l'année 1998, ainsi ...	Contribuable	Yes
437	284799	21/03/2008	Decision	Conseil d'État N° 284799 ECLI:FR:CESJS:2008:284799.20080321 Inédit au recueil Lebon 8ème sous-section jugeant seule M. Le Roy, président M. Patrick Quinqueton, rapporteur Mme Escaut Nathalie, commissaire du gouvernement SCP WAQUET, FARGE, HAZAN, avocats Lecture du vendredi 21 mars 2008	Vu la requête sommaire et le mémoire complémentaire, enregistrés les 6 septembre 2005 et 6 janvier 2006 au secrétariat du contentieux du Conseil d'Etat, présentés pour M. Louis A, demeurant ...; M. A demande au Conseil d'Etat : 1°) d'annuler l'arrêt du 7 juillet 2005 par lequel la cour administr...	Article 1er : La requête de M. A est rejetée. Article 2 : La présente décision sera notifiée à M. Louis A et au ministre du budget, des comptes publics et de la fonction publique.	Administration	Yes

4.

DATA REFINEMENT

Data refinement

Here, our code filters out rows with undetermined outcomes, as well as identifying duplicate rows.

3	89bx00231	2/07/1990	CAA	Cour administrative d'appel de Bordeaux N° 89BX00231 Mentionné au tables du recueil Lebon 1E CHAMBRE M. Tourdias, président M. Piot, rapporteur M. de Malafosse, commissaire du gouvernement Lecture du 2 juillet 1990	Vu la décision en date du 1er décembre 1988, enregistrée au greffe de la cour le 15 décembre 1988, par laquelle le président de la 7ème sous-section de la Section du contentieux du Conseil d'Etat a transmis à la cour, en application de l'article 17 du décret n° 88-906 du 2 septembre 1988, la req...	Not found	Indéterminé	No

```
# Ensure full text display without truncation
# Filter the DataFrame to show only rows where 'Outcome' is 'Indéterminé'
df_indeterminate = df[df['Outcome'] == 'Indéterminé']
df_indeterminate
```

```
# Exclude indertermined outcome rows
df = df[df['Outcome'] != 'Indéterminé']
```

```
# Check for duplicate rows based on all columns
duplicates = df[df.duplicated()]

# Display the duplicates
print(duplicates)
```

```
Empty DataFrame
Columns: [Case Number, Date, Case, Participant Text, Body Text, Decision Text, Outcome, Avocat]
Index: []
```

```
# Check duplicate for case number column
duplicates_specific = df[df.duplicated(subset=['Case Number'])]

# Display the duplicates based on specific columns
print(duplicates_specific)
```

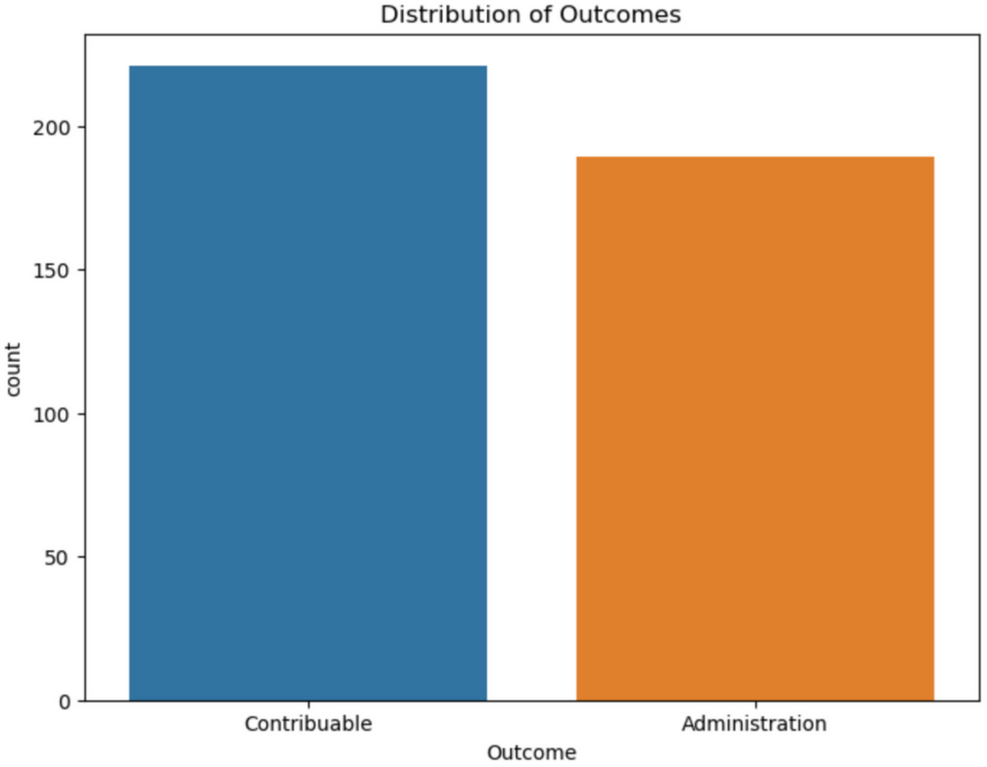
```
Empty DataFrame
Columns: [Case Number, Date, Case, Participant Text, Body Text, Decision Text, Outcome, Avocat]
Index: []
```

5. DATA ANALYSIS

Data analysis

Based on our results, we obtain a distribution of the cases won by the taxpayer and the tax authorities.

```
# DF contains only 'Contribuable' and 'Administration' outcomes
plt.figure(figsize=(8, 6))
sns.countplot(x='Outcome', data=df)
plt.title('Distribution of Outcomes')
plt.show()

# We have a close number of Contribuable and Administration,
# which is good for data analysis
```



The French Tax Authorities have had varying degrees of success when applying the abuse of law procedure (Article L.64 of the French Tax Procedures Code) in making adjustments to taxpayers.

The success rate can be inferred from the outcomes of cases where the presence of lawyers ('*Avocat*') is noted and whether the decision favored the administration or the taxpayer ('*Contribuable*').

Data analysis

A cross-tabulation analysis shows that in cases **without lawyers**, the **administration** won **83 times**, while **taxpayers** won **106 times**

In contrast, when lawyers were present, the **administration** won **37 times**, and **taxpayers** won **184 times.**

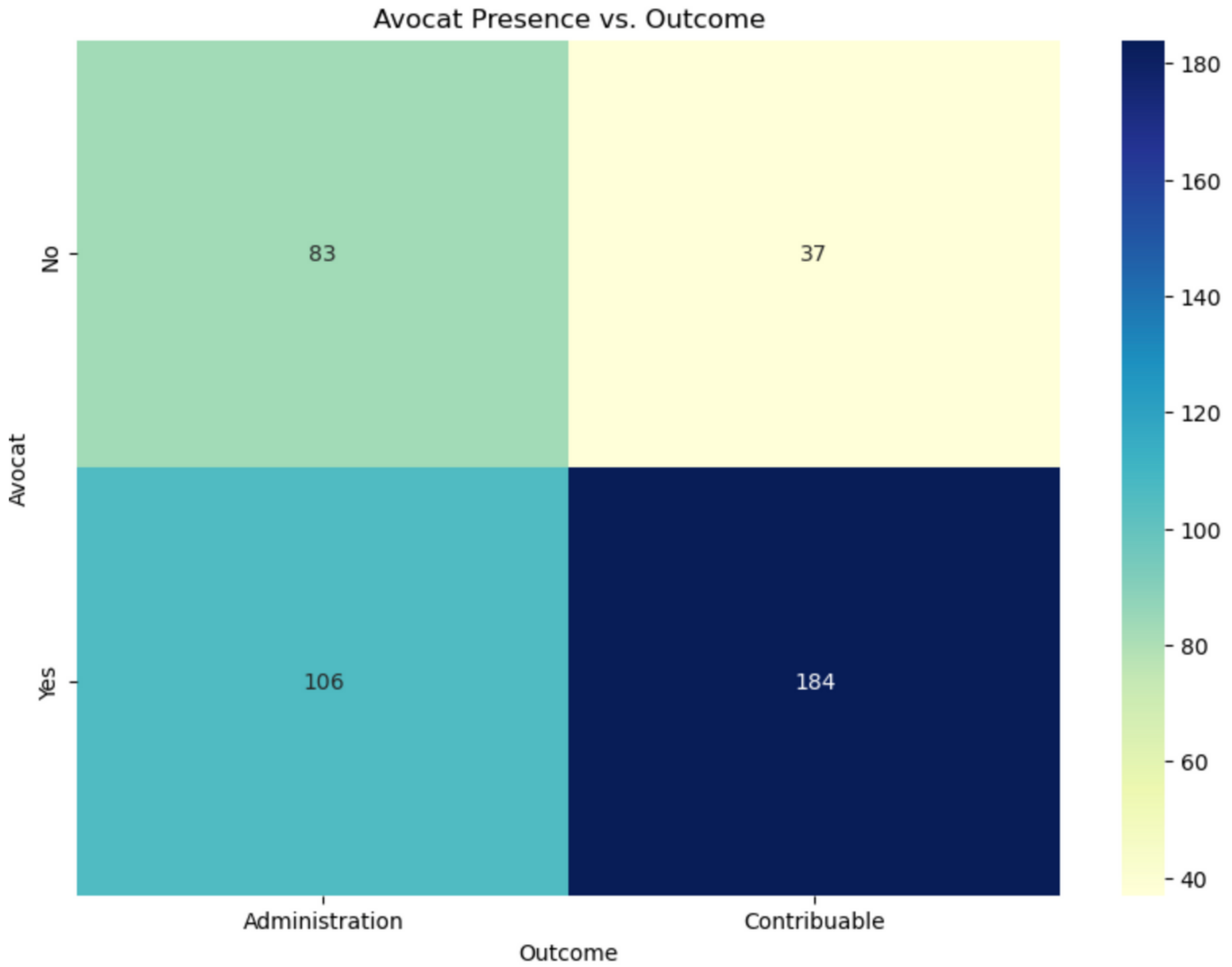
This suggests that **taxpayers with legal representation have a higher success rate** in disputes involving the abuse of law procedure.

```python
# Perform Chi-Square and cross tabulation to find if having an avocat can inscrease the change of winning
# Creating a cross-tabulation
cross_tab = pd.crosstab(df['Avocat'], df['Outcome'])

# Chi-Square Test
chi2, p_value, dof, expected = chi2_contingency(cross_tab)

# Displaying the cross-tabulation
print("Cross-tabulation:\n", cross_tab)
print("\nChi2 Statistic:", chi2, "\nP-value:", p_value)

# Plotting
plt.figure(figsize=(10, 7))
sns.heatmap(cross_tab, annot=True, cmap="YlGnBu", fmt="d")
plt.title('Avocat Presence vs. Outcome')
plt.ylabel('Avocat')
plt.xlabel('Outcome')
plt.show()

# Bar plot for visual comparison
cross_tab.plot(kind='bar', figsize=(10, 7))
plt.title('Avocat Presence vs. Outcome')
plt.ylabel('Count')
plt.xlabel('Avocat')
plt.xticks(rotation=0)
plt.show()

#p-value < 0.05, there is a significant result of the test
#Having an avocat inscrease the chance of winning
```

```
Cross-tabulation:
 Outcome  Administration  Contribuable
Avocat
No                   83            37
Yes                 106           184

Chi2 Statistic: 35.03569044770871
P-value: 3.2371714500784907e-09
```

Data analysis

We look at whether the presence of a lawyer affects the taxpayer's chances of winning.




```
Cross-tabulation:
 Outcome  Administration  Contribuable
Avocat
No                   83            37
Yes                 106           184

Chi2 Statistic: 35.03569044770871
P-value: 3.2371714500784907e-09
```

Data analysis

We look at whether the presence of a lawyer affects the taxpayer's chances of winning.

The **chi-square test** results from the analysis indicate a significant association between the presence of lawyers and the outcome categories.

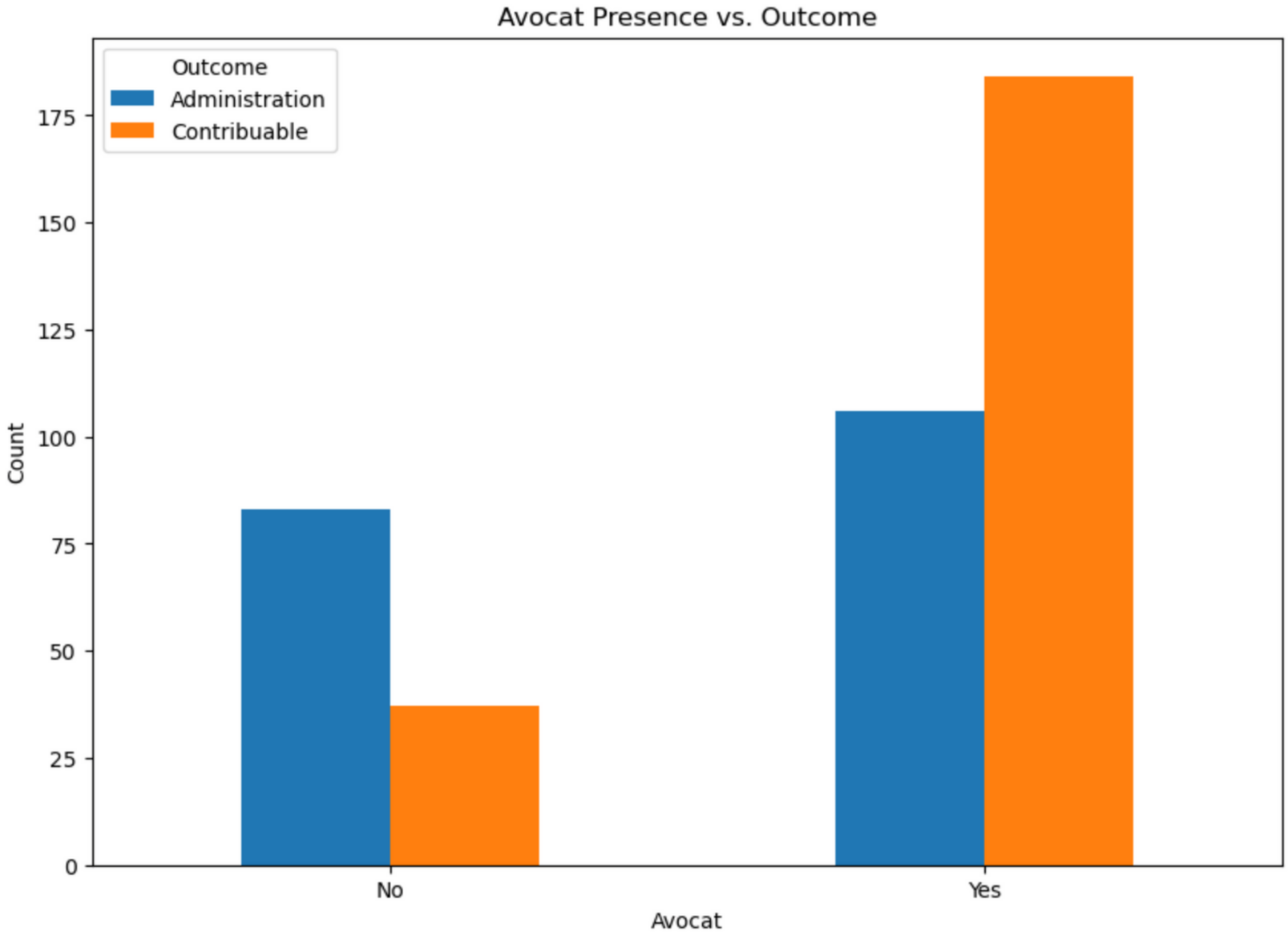
This low **p-value** suggests that legal representation may influence the success of taxpayers in these cases.

```
Cross-tabulation:
 Outcome  Administration  Contribuable
Avocat
No                    83            37
Yes                  106           184

Chi2 Statistic: 35.03569044770871
P-value: 3.2371714500784907e-09
```

Data analysis

This bar chart comparing the counts of cases for each outcome category based on the presence or absence of lawyers further illustrates the correlation between legal representation and different outcomes.

It shows that taxpayers are more likely to win their cases against the French Tax Authorities when they have legal representation.



In summary, the French Tax Authorities experience a **lower success rate** in cases where taxpayers are **represented by lawyers**, as indicated by the higher number of taxpayer wins in such cases.

The significant association between the presence of lawyers and favorable outcomes for taxpayers suggests that **legal representation plays a crucial role** in the application of the abuse of law procedure by the French Tax Authorities.

6. CHALLENGES ENCOUNTERED

Main challenges

Find the html and XPath zone associated with each element (search box, checkboxes, etc.).

Solution: take a close look at the html codes via *inspect* and find the associated attributes.
For Xpath, ask Chat GPT.

Creation of the dataframe.

E.g., the creation of the date column that requires to find the associated pattern and convert to format (French => English).
Solution: create a dictionary that allows to translate the month (for example January = 01) so that it can be parsed by Python.

Find 'decision' patterns.

Solution: test several keywords and keep the ones that work well (*i.e.* that makes it possible to determine who wins at the end of the procedure).

Thanks
for listening!