

ANALYSIS OF THE REVERSAL RATE ON APPEAL OF COMMERCIAL COURT JUDGMENTS IMPOSING SANCTIONS IN INSOLVENCY LAW (FAILLITE PERSONNELLE AND INTERDICTION DE GÉRER)

BY CHLOÉ GOLDSTEIN, CAROLINE PROT, ELVIRE JOLIMOY AND JULIEN LAPORTE

STEP 1: INTRODUCTION

STEP 2: CREATION OF THE DATA FRAME

STEP 3: ANALYSIS OF THE DATA FRAME

STEP 4: CONCLUSION

SUMMARY



STEP I: INTRODUCTION

INTRODUCTION

I. DEFINITIONS OF PERSONAL BANKRUPTCY AND MANAGEMENT BAN: The sanction of personal bankruptcy may be pronounced by the court in charge of the collective proceedings when the manager has committed a fault in the management of his company, thus participating in the opening of a judicial recovery or liquidation procedure.

- Personal bankruptcy (*faillite personnelle*): « *Personal bankruptcy entails a ban on directing, managing, administering or controlling, directly or indirectly, any commercial or craft enterprise, any farm or any enterprise with any other independent activity and any legal person* » (Article L.653-2 of the French Commercial Code)
- Management ban (*interdiction de gérer*): ban on directing, managing, administering or controlling, directly or indirectly, a company or any legal person (Article L.653-8 of the French Commercial Code)



II. PURPOSE OF THE PROJECT: to analyze the number of decisions sentencing to such sanctions, the rate of confirmation in appeal and comparing the severity of the different Courts.

STEP 2: CREATION OF THE DATA FRAME

PART I: IMPORTATION OF THE PACKAGES NECESSARY

```
1 # PARTIE 1 : Importation des différents packages nécessaires
2 import regex as re
3 import requests
4 from bs4 import BeautifulSoup
5 from selenium.webdriver.common.by import By
6 from selenium import webdriver
7 from selenium.webdriver.chrome.service import Service
8 from webdriver_manager.chrome import ChromeDriverManager
9 from selenium.webdriver.common.keys import Keys as KeysBrowser
10 import pandas as pd
11 import time
12 from matplotlib import pyplot as plt
13 import openpyxl
14 from pathlib import Path
```

PART 2.I: SCRAPING OF DECISIONS WITH « FAILLITE PERSONNELLE »

SUBPART I to 5: Looking for decisions on the Cour de cassation website

```
# 2.1 : Extraction des décisions avec "Faillite personnelle"
```

1 1.1 : Accès au site de recherche de la Cour de cassation

```
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
driver.get("https://www.courdecassation.fr/acces-rapide-judilibre") # Accès au site
time.sleep(2) # Attente pour chargement de la page
```

2 1.2 : Ecriture dans la barre de recherche

```
att_texte = "edit-search-api-fulltext" # Repérage de la barre de recherche
texte = driver.find_element(By.XPATH, r'//*[@@data-drupal-selector="' + att_texte + '"]')
texte.send_keys("Faillite personnelle") # Ecriture dans barre de recherche
rep_faill = "RepDecisions_FaillitePersonnelle/" # Nom du dossier où s'enregistreront la liste
rep = rep_faill
```

3 1.3 : Click dans la case "Expression exacte"

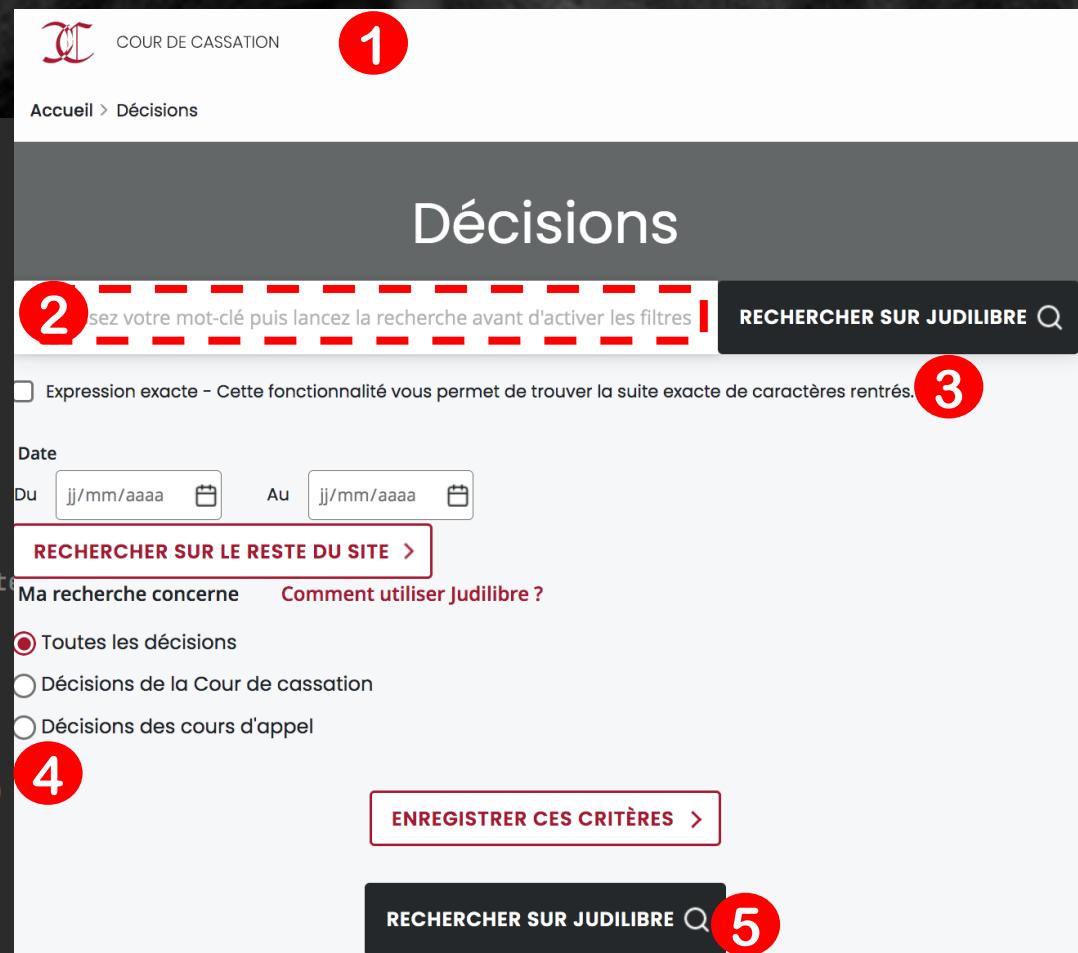
```
att_box = "edit-expression-exacte" # Repérage de la case
box = driver.find_element(By.XPATH, r'//*[@@data-drupal-selector="' + att_box + '"]')
box.click() # Click sur la case
```

4 1.4 : Click dans la case "Décisions de cours d'appel"

```
att_circle = "edit-judilibre-jurisdiction-ca"
circle = driver.find_element(By.XPATH, r'//*[@@data-drupal-selector="' + att_circle + '"]')
circle.click()
```

5 1.5 : Click dans la case "Rechercher"

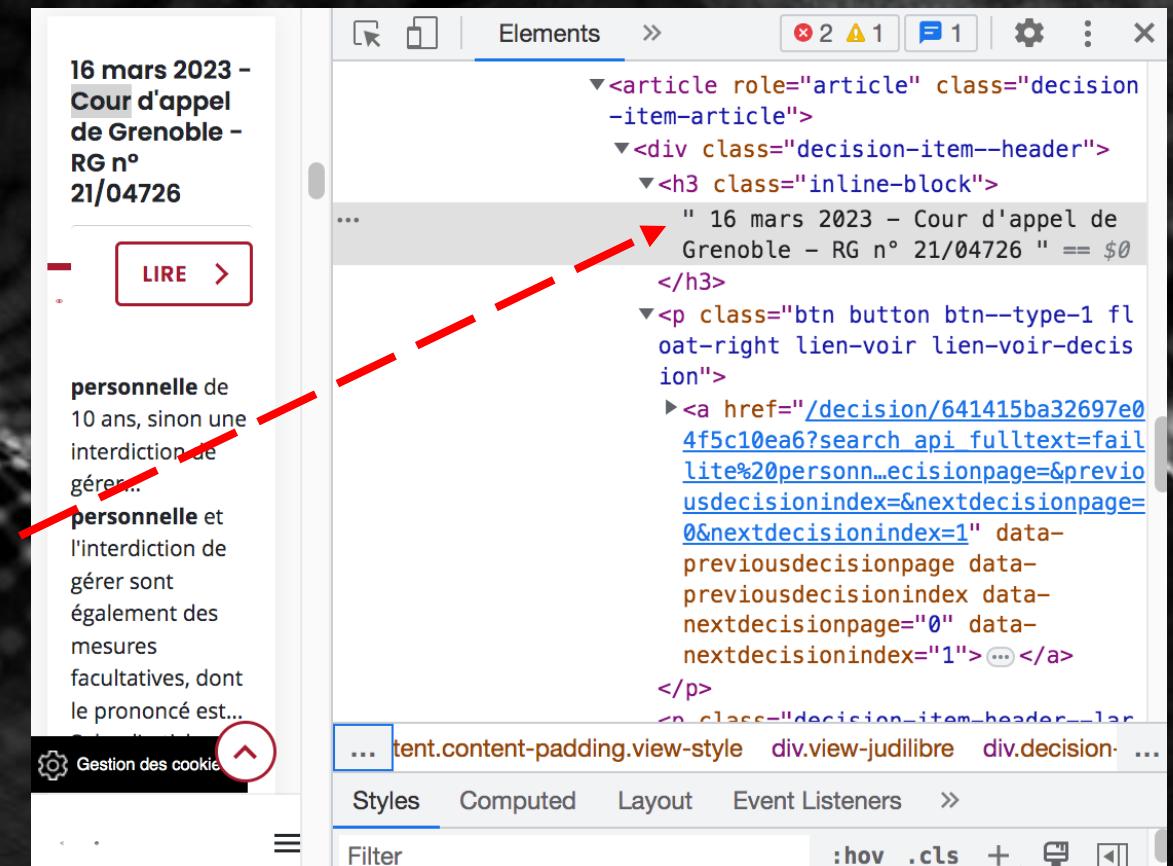
```
att_rechercher = "edit-submit-judilibre"
rechercher = driver.find_element(By.XPATH, r'//*[@@data-drupal-selector="' + att_rechercher + '"]')
rechercher.click()
```



PART 2.I: SCRAPING OF DECISIONS WITH « FAILLITE PERSONNELLE »

SUBPART 6.I: File name cleanup

```
# 2.1.6 : Récupération de chaque décision sur toutes les pages
site_cass = "https://www.courdecassation.fr"
# Enregistrement dans une variable du début de l'adresse du site
fin = False
# Variable pour arrêter la boucle While lorsqu'il y a plus de page
while fin == False :
    print ("je consulte la page " + str(counter))
    # Utile lors du débug pour voir la page consultée
    soup = BeautifulSoup(driver.page_source)
    # Repérage du lieu avec les informations qui
    # seront contenues dans le nom du fichier
    decision_div = soup.find_all("div", class_="decision-item--header")
    # Nettoyage du nom du fichier
    for div in decision_div :
        h3_div = div.find_next(["h3"])
        t = h3_div.text
        t = t.replace("\n", "")
        t = t.replace("\t", "")
        t = t.replace("°", "_")
        t = t.replace("/", "_")
        t = t.replace("'", " ")
        fic = t + ".txt" # Nom du fichier
```

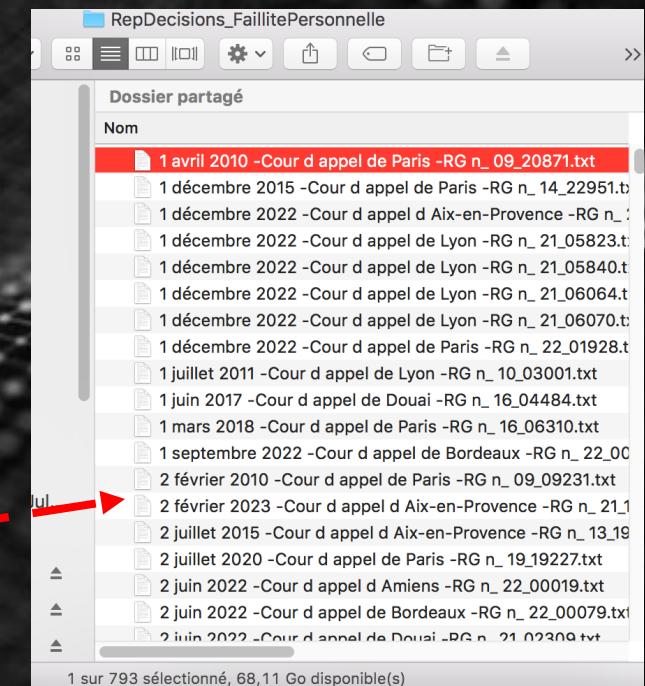


PART 2.I: SCRAPING OF DECISIONS WITH « FAILLITE PERSONNELLE »

SUBPART 6.2: Creation and name writing for every file

```
# Vérification que le fichier n'existe pas
# déjà s'il y a deux lignes pour la même décision
if os.path.isfile(rep + fic) :
    print("le fichier existe : " + fic)
# Sinon récupération du texte de la décision
else :
    a_div = div.find_next(["a"])
    # Récupération du lien
    lien = a_div.get('href')
    webpage_decision = requests.get(site_cass + lien)
    # Accès à la page de la décision
    time.sleep(2) # Temps de chargement de la page
    tmp_file_name = rep + "tmp"
    # Création du fichier txt dans répertoire
    tmp_file = open(tmp_file_name, "w")
    # Ouverture du fichier
    tmp_file.write(webpage_decision.text)
    # Ecriture dans fichier du text de la décision
    tmp_file.close() # Fermeture du fichier
    os.rename(tmp_file_name, rep + fic)
    # Changement du nom du fichier
```

A screenshot of a terminal window titled "1 avril 2010 -Cour d appel de Paris -RG n_09_20871.txt". The window displays the HTML code of a court decision. Several lines of code are highlighted with red arrows pointing from the left margin to specific parts of the code, likely indicating the extraction logic. The code includes details such as the date (01 AVRIL 2010), the court (COUR D'APPEL DE PARIS), the chamber (5ème Chambre), and the case number (09/20871).



PART 2.I: SCRAPING OF DECISIONS WITH « FAILLITE PERSONNELLE »

SUBPART 6.3: Finding the link and going to the next page

```
list_url = soup.find_all("a")
# Recherche de tous les liens dans la page
page_suivante = ""
for x in list_url:
    lien = x.get('href')
    # Recherche du lien permettant d'accéder
    # à la page suivante
    if lien is not None:
        if lien.startswith("/recherche") :
            title = x.get('title')
            if title == "Aller à la page suivante":
                page_suivante = lien
if page_suivante != "":
    # Si existence du lien, accès à la page suivante
    driver.get(site_cass + page_suivante)
    time.sleep(1)
else:
    # Si pas de lien pour page suivante,
    # alors arrêt de la boucle While
fin = True
```

The screenshot shows a web browser window with a dark theme. On the left, the code for scraping decisions is displayed. In the center, a court decision page from the Cour d'appel de Lyon is shown. The page title is "9 mars 2023 - Cour d'appel de Lyon - RG n° 20/06317". It contains text about a judgment from August 4, 2020, regarding personal bankruptcy. At the bottom right of the page, there is a navigation bar with links like "Aller à la page suivante" and "Dernière »". A red arrow points from the word "page suivante" in the code to the "Aller à la page suivante" link on the page. Another red arrow points from the "Dernière »" link on the page to the "next" link in the developer tools' element inspector. On the right, the developer tools' element inspector is open, showing the HTML structure of the page. A blue highlight is applied to the "next" link in the navigation bar's HTML code, which corresponds to the "Aller à la page suivante" link on the page itself.

```
> <div class="decision-item">...</div>
> <div class="decision-item">...</div>
> <div class="decision-item">...</div>
> <div class="decision-item">...</div>
</div>
<nav class="pager" role="navigation" aria-label="Pagination">
  <ul class="pager__items js-pager__items">
    <li class="pager__item is-active">...</li>
    <li class="pager__item">...</li>
    <li class="pager__item pager__item--next">
      <a href="/recherche-judilibre?search_api_full_text=faillite%20personnelle&expression.=&date_au=&judilibre_jurisdiction=ca&op=Rechercher%20sur%20judilibre&page=1" title="Aller à la page suivante" rel="next">...</a> == $0
    </li>
    <li class="pager__item pager__item--last">...</li>
  </ul>
  ... ul.pager__items.js-pager__items li.pager__item.pager__item--next a ...
```

PART 2.2: SCRAPING OF DECISIONS WITH « INTERDICTION DE GÉRER »

```
# 2.2 : Extraction des décisions avec "Interdiction de gérer"

# 2.2.1 : Accès au site de recherche de la Cour de cassation
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
driver.get("https://www.courdecassation.fr/acces-rapide-judilibre")
time.sleep(2)

# 2.2.2 : Ecriture dans la barre de recherche
att_texte = "edit-search-api-fulltext"
texte = driver.find_element(By.XPATH, r'.//*[@@data-drupal-selector="' + att_texte + '"]')
texte.send_keys("Interdiction de gérer")
rep_inter = "RepDecisions_InterdictionDeGerer/"
rep = rep_inter

# 2.2.3 : Click dans la case "Expression exacte"
att_box = "edit-expression-exacte"
box = driver.find_element(By.XPATH, r'.//*[@@data-drupal-selector="' + att_box + '"]')
box.click()

# 2.2.4 : Click dans la case "Décisions de cours d'appel"
att_circle = "edit-judilibre-juridiction-ca"
circle = driver.find_element(By.XPATH, r'.//*[@@data-drupal-selector="' + att_circle + '"]')
circle.click()

# 2.2.5 : Click dans la case "Rechercher"
att_rechercher = "edit-submit-judilibre"
rechercher = driver.find_element(By.XPATH, r'.//*[@@data-drupal-selector="' + att_rechercher + '"]')
rechercher.click()
```

```
# 2.2.6 : Récupération de chaque décision sur toutes les pages
site_cass = "https://www.courdecassation.fr"
fin = False
counter = 1

while fin == False :
    print ("je consulte la page " + str(counter))
    counter += 1
    soup = BeautifulSoup(driver.page_source)
    decision_div = soup.find_all("div", class_="decision-item--header")
    for div in decision_div :
        h3_div = div.find_next(["h3"])
        t = h3_div.text
        t = t.replace("\n", " ")
        t = t.replace("\t", " ")
        t = t.replace("°°", "_")
        t = t.replace("/", "_")
        t = t.replace("//", " ")
        fic = t + ".txt"
        if os.path.isfile(rep + fic) :
            print("le fichier existe : " + fic)
        else :
            a_div = div.find_next(["a"])
            lien = a_div.get('href')
            webpage_decision = requests.get(site_cass + lien)
            time.sleep(2)
            tmp_file_name = rep + "tmp"
            tmp_file = open(tmp_file_name, "w")
            tmp_file.write(webpage_decision.text)
            tmp_file.close()
            os.rename(tmp_file_name, rep + fic)
    list_url = soup.find_all("a")
    page_suivante = ""
    for x in list_url :
        lien = x.get('href')
        if lien is not None :
            if lien.startswith("/recherche") :
                title = x.get('title')
                if title == "Aller à la page suivante" :
                    page_suivante = lien
    if page_suivante != "" :
        driver.get(site_cass + page_suivante)
        time.sleep(1)
    else :
        fin = True
```

PART 3.I:ANALYSIS OF FILES WITH « FAILLITE PERSONNELLE »

SUBPART I: Finding data : Date, CA and Numero of decisions

```
# PARTIE 3 : Analyse des fichiers
# 3.1 : Analyse des fichiers contenant "Faillite personnelle"
erreurs = []

# Variable utile pour repérer les décisions
# ne comportant pas les mots recherchés
rep = rep_faill # Accès au bon dossier
list_of_lists = []

# Liste des listes permettant de former le Data Frame

# Ouverture de chaque fichier du dossier
for file in os.listdir(rep):
    if file.endswith(".txt"):
        fic = open(rep + file, "r")
        contenu = fic.read()
        fic.close()
        soup = BeautifulSoup(contenu, "html.parser")
        # Conversion du fichier txt en soup
        decision_div = soup.find("div", class_="decision-content decision-content--main")

        # Recherche du titre avec les infos voulues (Date, CA, Numéro)
        divs = decision_div.find_all_next(["h1"])
        if len(divs) > 1: # Vérification qu'il n'y a qu'un titre
            print("STOP car plus d'un H1 dans le fichier " + file + "(" + len(divs)+ ")")
            exit()

        for sib in decision_div.find_all_next(["h1"]):
            liste = str(sib).split("<br/>")
            # Nettoyage des informations
            for i in range(0, len(liste)):
                liste[i] = liste[i].replace("<h1>", "")
                liste[i] = liste[i].replace("\n", "")
                liste[i] = liste[i].replace("\t", "")
                liste[i] = liste[i].replace("</h1>", "")
            # Dans l'ordre liste : Date, CA, Numéro
            list_of_lists.append(liste)
```



PART 3.I:ANALYSIS OF FILES WITH « FAILLITE PERSONNELLE »

SUBPART 2: Finding the ruling part

Step I. Finding « Par ces motifs »

```
# Recherche du verdict du jugement
jugement_div = soup.find_all("div", class_="decision-content decision-content--main")[-1]
jugement = ""

# Recherche de la position du "Par ces motifs" pour chercher le verdict après
# Synonyme de Par ces motifs : M O T I F et Statuant publique
position_PCM = jugement_div.text.upper().find("PAR CES MOTIFS")
if position_PCM == -1:
    position_PCM = jugement_div.text.upper().find("M O T I F")
    if position_PCM == -1 :
        position_PCM = jugement_div.text.upper().find("STATUANT PUBLIQUEMENT")

# Vérification de l'existence d'un "Par ces motifs" dans la décision
if position_PCM == -1:
    print("pas de PCM dans " + file)
```

```
# Recherche de la position du verdict d'infirmer
# Synonyme d'infirmer : infime, infirmons, annule, annulons, annulation, réformant, réforme, suspension,
# suspend, arrêt de l'exécution, arrêtons l'exécution, arrêter l'exécution, nul et de nul effet le jugement,
# nullité du jugement.
position_infirme = jugement_div.text.upper().find("INFIRME",position_PCM)
if position_infirme == -1:
    position_infirme = jugement_div.text.upper().find("INFIRMONS",position_PCM)
    if position_infirme == -1:
        position_infirme = jugement_div.text.upper().find("ANNULE",position_PCM)
        if position_infirme == -1 :
            position_infirme = jugement_div.text.upper().find("ANNULONS",position_PCM)
            if position_infirme == -1:
                position_infirme = jugement_div.text.upper().find("ANNULATION", position_PCM)
                if position_infirme == -1:
                    position_infirme = jugement_div.text.upper().find("REFORMANT",position_PCM)
                    if position_infirme == -1 :
                        position_infirme = jugement_div.text.upper().find("RÉFORMANT",position_PCM)
                        if position_infirme == -1:
                            position_infirme = jugement_div.text.upper().find("REFORME",position_PCM)
                            if position_infirme == -1:
                                position_infirme = jugement_div.text.upper().find("RÉFORME",position_PCM)
                                if position_infirme == -1:
                                    position_infirme = jugement_div.text.upper().find("SUSPENSION",position_PCM)
                                    if position_infirme == -1:
                                        position_infirme = jugement_div.text.upper().find("SUSPEND",position_PCM)
                                        if position_infirme == -1:
                                            position_infirme = jugement_div.text.upper().find("ARRÊT DE L'EX",position_PCM)
                                            if position_infirme == -1:
                                                position_infirme = jugement_div.text.upper().find("ARRET DE L'EX",position_PCM)
                                                if position_infirme == -1:
                                                    position_infirme = jugement_div.text.upper().find("ARRÊTONS L'EX", position_PCM)
                                                    if position_infirme == -1:
                                                        position_infirme = jugement_div.text.upper().find("ARRETONS L'EX", position_PCM)
                                                        if position_infirme == -1:
                                                            position_infirme = jugement_div.text.upper().find("ARRÊTER L'EX", position_PCM)
                                                            if position_infirme == -1:
                                                                position_infirme = jugement_div.text.upper().find("ARRETER L'EX", position_PCM)
                                                                if position_infirme == -1:
                                                                    position_infirme = jugement_div.text.upper().find("NUL ET DE NUL EFFET LE JUGEMENT",position_PCM)
                                                                    if position_infirme == -1:
                                                                        position_infirme = jugement_div.text.upper().find("NULLITE DU JUGEMENT",position_PCM)
                                                                        if position_infirme == -1:
                                                                            position_infirme = jugement_div.text.upper().find("NULLITÉ DU JUGEMENT",position_PCM)
```

Step 2. Finding if the decision is invalidated?

Synonyms: *infirme, infirmons, annule, annulons, annulation, réformant, réforme, suspension, suspend, arrêt de l'exécution, arrêter l'exécution, nul et de nul effet, nullité*

PART 3.I:ANALYSIS OF FILES WITH « FAILLITE PERSONNELLE »

SUBPART 2: Finding the ruling

Step 3. Finding if the decision is validated?

```
# Recherche de la position du verdict de confirmer
# Synonyme d'infirmer : confirme, confirmons, rejettors, rejette, erreur matérielle, déboute, déboutons,
# écarte, écartons, rectification
position_confirme = jugement_div.text.upper().find("CONFIRME",position_PCM)
if position_confirme == -1:
    position_confirme = jugement_div.text.upper().find("CONFIRMONS",position_PCM)
    if position_confirme == -1:
        position_confirme = jugement_div.text.upper().find("REJETONS",position_PCM)
        if position_confirme == -1:
            position_confirme = jugement_div.text.upper().find("REJETTE",position_PCM)
            if position_confirme == -1:
                position_confirme = jugement_div.text.upper().find("ERREUR MATÉRIELLE",position_PCM)
                if position_confirme == -1:
                    position_confirme = jugement_div.text.upper().find("DÉBOUTE",position_PCM)
                    if position_confirme == -1:
                        position_confirme = jugement_div.text.upper().find("DÉBOUTONS",position_PCM)
                        if position_confirme == -1:
                            position_confirme = jugement_div.text.upper().find("ÉCARTE",position_PCM)
                            if position_confirme == -1:
                                position_confirme = jugement_div.text.upper().find("ÉCARTONS",position_PCM)
                                if position_confirme == -1:
                                    position_confirme = jugement_div.text.upper().find("RECTIFICATION",position_PCM)
```

Synonyms: *confirme, confirmons, rejettors, rejette, erreur matérielle, déboute, déboutons, écarte, écartons, rectification*

Step 4. Finding if the decision is inadmissible, irrelevant, null and void or reopening of debats?

```
# Recherche de la position de Irrecevable
position_irrecevable = jugement_div.text.upper().find("IRRECEVABLE",position_PCM)

# Recherche de la position de Sans objet
# Synonyme de Sans objet : saisie d'aucun, désistement, radiation
position_sansobjet = jugement_div.text.upper().find("SANS OBJET",position_PCM)
if position_sansobjet == -1 :
    position_sansobjet = jugement_div.text.upper().find("SAISIE D'AUCUN", position_PCM)
    if position_sansobjet == -1:
        position_sansobjet = jugement_div.text.upper().find("DÉSISTEMENT", position_PCM)
        if position_sansobjet == -1:
            position_sansobjet = jugement_div.text.upper().find("DESISTEMENT", position_PCM)
            if position_sansobjet == -1:
                position_sansobjet = jugement_div.text.upper().find("RADIATION", position_PCM)

# Recherche de la position de Caduque
# Synonyme de Caduque : caducité
position_caduque = jugement_div.text.upper().find("CADUQUE",position_PCM)
if position_caduque == -1:
    position_caduque = jugement_div.text.upper().find("CADUCITÉ", position_PCM)
    if position_caduque == -1:
        position_caduque = jugement_div.text.upper().find("CADUCITE", position_PCM)

# Recher de la position de Reouverture des débats
# Synonyme de Réouverture des débats : désistement d'appel
position_reouverture = jugement_div.text.upper().find("OUVERTURE DES DÉBATS",position_PCM)
if position_reouverture == -1 :
    position_reouverture = jugement_div.text.upper().find("OUVERTURE DES DEBATS",position_PCM)
    position_desistement = jugement_div.text.upper().find("DÉSISTEMENT D'APPEL",position_PCM)
```

PART 3.I:ANALYSIS OF FILES WITH « FAILLITE PERSONNELLE »

SUBPART 2: Finding the ruling

Step 5. Comparing positions of the terms

```
# Comparaison des positions
if position_infirme == -1 : # Si pas d'infirmination
    if position_confirme == -1 : # Si pas de confirmation
        if position_irrecevable == -1 : # Si pas d'irrecevabilité
            if position_sansobjet == -1 : # Si pas de sans objet
                if position_reouverture == -1 : # Si pas de réouverture
                    if position_desistement == -1 : # Si pas de désistement
                        if position_caduque == -1 : # Si pas de caduque
                            # Ajout du fichier à la liste d'erreur pour aller trouver le terme utilisé
                            print("Pas de confirmation ni d'annulation ni d'irrecevable dans " + file)
                            erreur.append(file)
                    else :
                        jugement = "Caduque"
                else :
                    jugement = "Désistement d'appel"
            else :
                jugement = "Réouverture des débats"
        else :
            jugement = "Sans objet"
    else :
        jugement = "Irrecevable"
else :
    jugement = "Confirmation totale"
# Comparaison des positions d'infirmination et de confirmation
# Si infirmination apparait avant confirmation alors infirmination partielle
# Si c'est l'inverse, confirmation partielle
else :
    if position_confirme == -1:
        jugement = "Infirmination totale"
    else :
        if position_confirme > position_infirme :
            jugement = "Infirmination partielle"
        else :
            jugement = "Confirmation partielle"
# Il y a 9 verdicts possibles : Confirmation totale, Infirmination totale, Confirmation partielle,
# Infirmination partielle, Caduque, Désistement d'appel, Réouverture des débats, Sans objet, Irrecevable
```

PART 3.I:ANALYSIS OF FILES WITH « FAILLITE PERSONNELLE »

SUBPART 3: Incrementing the list of lists

```
liste.append(jugement) # ordre de la liste : Date, CA, Numéro, Verdict  
liste.append("1") # Rajout à la liste de l'information que cette décision vient du dossier Faillite personnelle  
liste.append("0") # Rajout de l'information que cette décision ne vient pas du dossier Interdiction de gérer  
liste.append(file) # Rajout du nom du fichier à la fin  
list_of_lists.append(liste) # Rajout de la liste à la liste de listes
```

PART 3.2:ANALYSIS OF FILES WITH « INTERDICTION DE GERER »

Before doing the same:

Check if the decision is already present in the other folder

```
# 3.2 : Analyse des fichiers contenant "Interdiction de gérer"
rep = rep_inter

if file.endswith(".txt"):

    # Vérification de la présence ou non de la décision dans le dossier Faillite personnelle
    for file in os.listdir(rep):
        if Path(rep_faill + file).is_file():
            # Si oui mettre 1 dans la colonne correspondante
            for list in list_of_lists:
                if list[-1] == file:
                    list[-2] = 1

    # Si non faire la même procédure que précédemment et extraction des informations de la décisions
else :
```

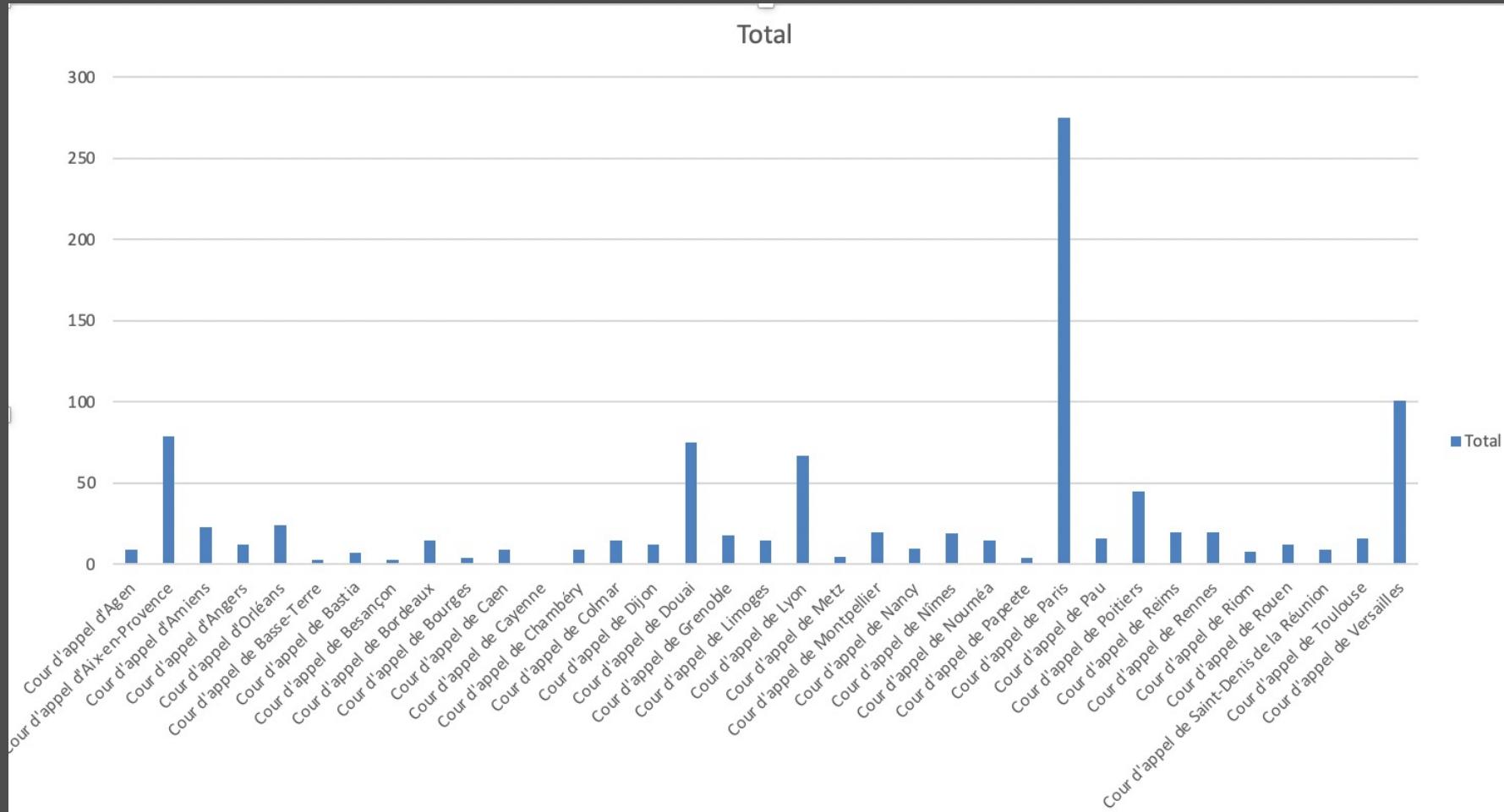
PART 4: EXPORTATION OF DATA IN EXCEL

```
# PARTIE 4 : Exportation dans un document Excel des données
df = pd.DataFrame(list_of_lists,columns=["Date","Ville de la Cour d'appel","Numéro","Verdict","Présence de \"faillite personnelle\"", "Présence de \"Interdiction de gérer\"", "Nom du fichier"])
df = df.drop(columns=['Nom du fichier']) # Suppression de la colonne nom du fichier
df.to_excel("Liste-DecisionsAvecInfos.xlsx") # Exportation dans un Excel
```

Date	Ville de la Cour d'appel	Numéro	Verdict	Verdict synthèse Confirmation si = 1	Présence de "Faillite personnelle"	Présence de "Interdiction de gérer"
23/02/2023	Cour d'appel d'Aix-en-Provence	RG n° 22/01026	Infirmation partielle	0	1	1
23/02/2023	Cour d'appel d'Aix-en-Provence	RG n° 22/13001	Confirmation totale	1	1	0
23/02/2023	Cour d'appel d'Aix-en-Provence	RG n° 22/02191	Confirmation totale	1	0	1
23/02/2023	Cour d'appel d'Orléans	RG n° 22/01519	Infirmation totale	0	0	1
23/02/2023	Cour d'appel de Grenoble	RG n° 21/02977	Infirmation partielle	0	0	1
23/02/2023	Cour d'appel de Paris	RG n° 22/13224	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13192	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13182	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13169	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13208	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13209	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13210	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13173	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13171	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13212	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 22/13217	Confirmation totale	1	1	0
23/02/2023	Cour d'appel de Paris	RG n° 23/02137	Infirmation partielle	0	1	0
23/02/2023	Cour d'appel de Pau	RG n° 22/02171	Infirmation partielle	0	1	0
22/02/2023	Cour d'appel de Paris	RG n° 21/13910	Infirmation partielle	0	1	1
22/02/2023	Cour d'appel de Paris	RG n° 21/09325	Confirmation partielle	1	1	1
22/02/2023	Cour d'appel de Paris	RG n° 20/14517	Infirmation partielle	0	1	1
22/02/2023	Cour d'appel de Paris	RG n° 22/08803	Confirmation totale	1	0	1
21/02/2023	Cour d'appel de Reims	RG n° 22/01669	Confirmation partielle	1	1	1
21/02/2023	Cour d'appel de Reims	RG n° 22/01628	Confirmation partielle	1	1	0
16/02/2023	Cour d'appel d'Amiens	RG n° 22/00234	Confirmation totale	1	0	1
16/02/2023	Cour d'appel de Metz	RG n° 22/00323	Infirmation partielle	0	1	1
16/02/2023	Cour d'appel de Pau	RG n° 22/00919	Infirmation partielle	0	1	0
15/02/2023	Cour d'appel de Bastia	RG n° 21/00895	Infirmation partielle	0	1	0
15/02/2023	Cour d'appel de Bastia	RG n° 21/00736	Infirmation totale	0	0	1

STEP 3: ANALYSIS OF THE DATA FRAME

NUMBER OF DECISIONS IMPOSING SANCTIONS IN INSOLVENCY LAW (FROM 1998 TO 2023)



KEY POINTS

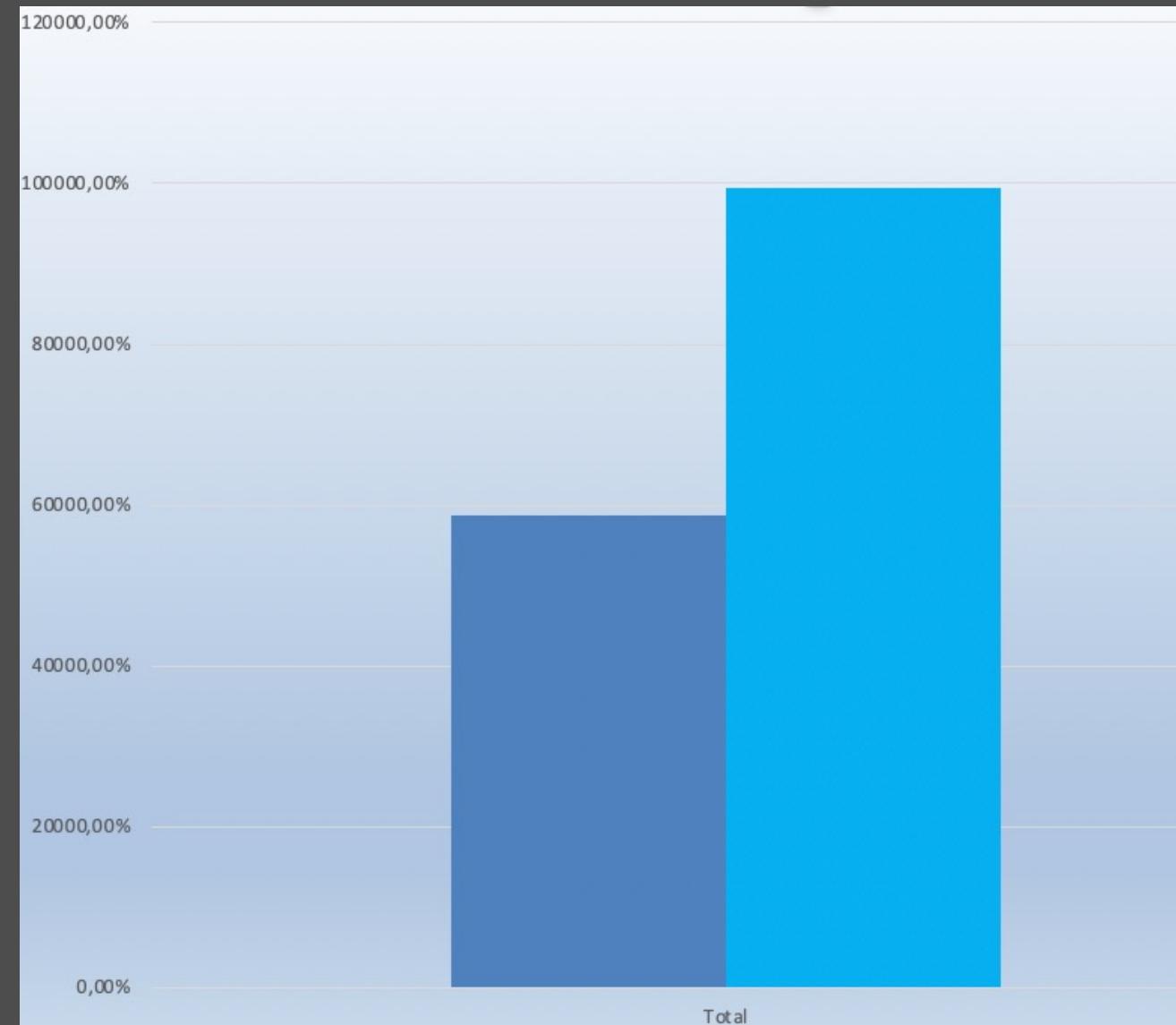
- Few examples:
 - Paris: 275
 - Versailles: 101
 - Cayenne: 1
 - Papeete: 4
- Total number of decisions: 995
- Courts of Appeal that impose the highest number of sanctions:
 - Paris
 - Versailles
 - Aix-en-Provence
 - Douai
 - Lyon

- Some Courts of appeal issue many decisions whereas others issue very few decisions
- Representative of the **economic distribution of wealth in France** and the economic dynamism of a geographical area

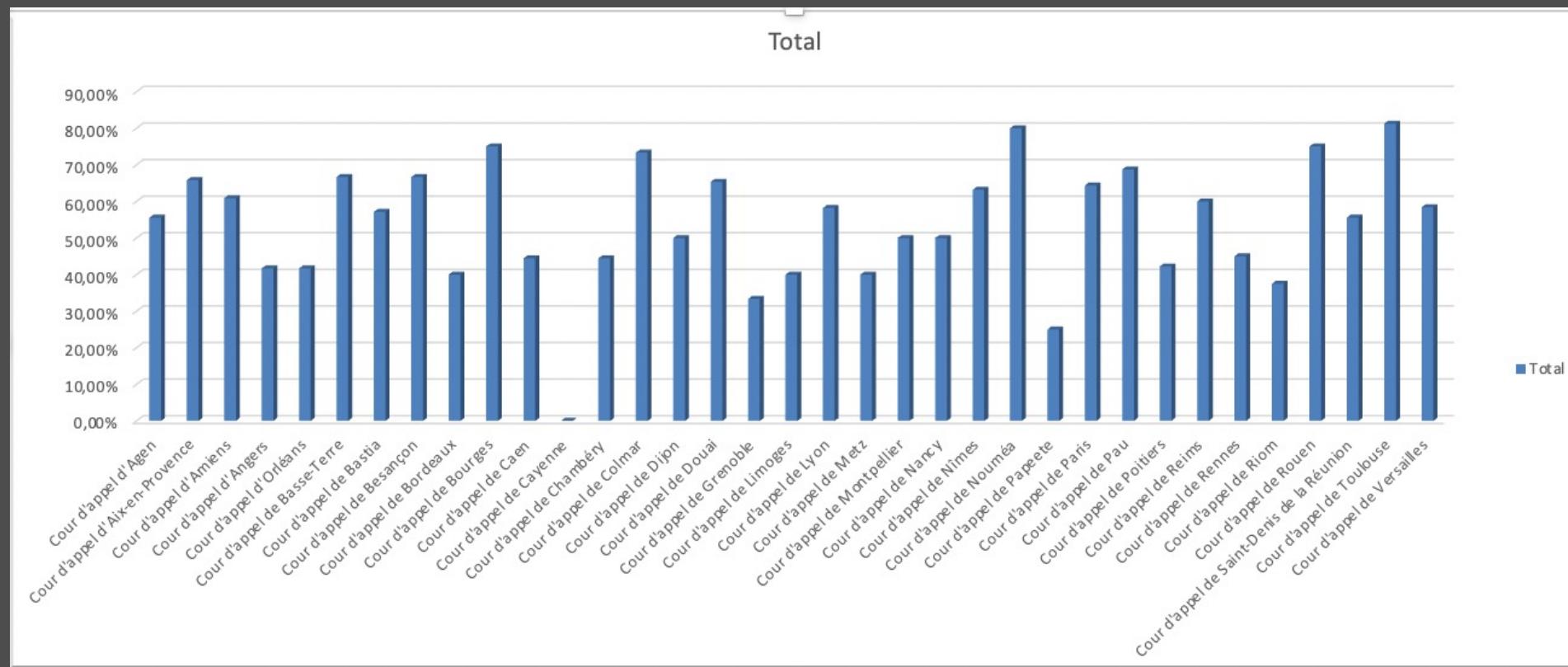
RATE OF SUCCESS (ALL COURTS OF APPEAL AND ALL YEARS)

KEY POINTS

- Percentage of first instance decisions imposing sanctions upheld on appeal: 58,7%
- Percentage of first instance decisions overturned on appeal: 41,3%
- Conclusion: a first instance decision imposing insolvency law sanction is more likely to be upheld on appeal



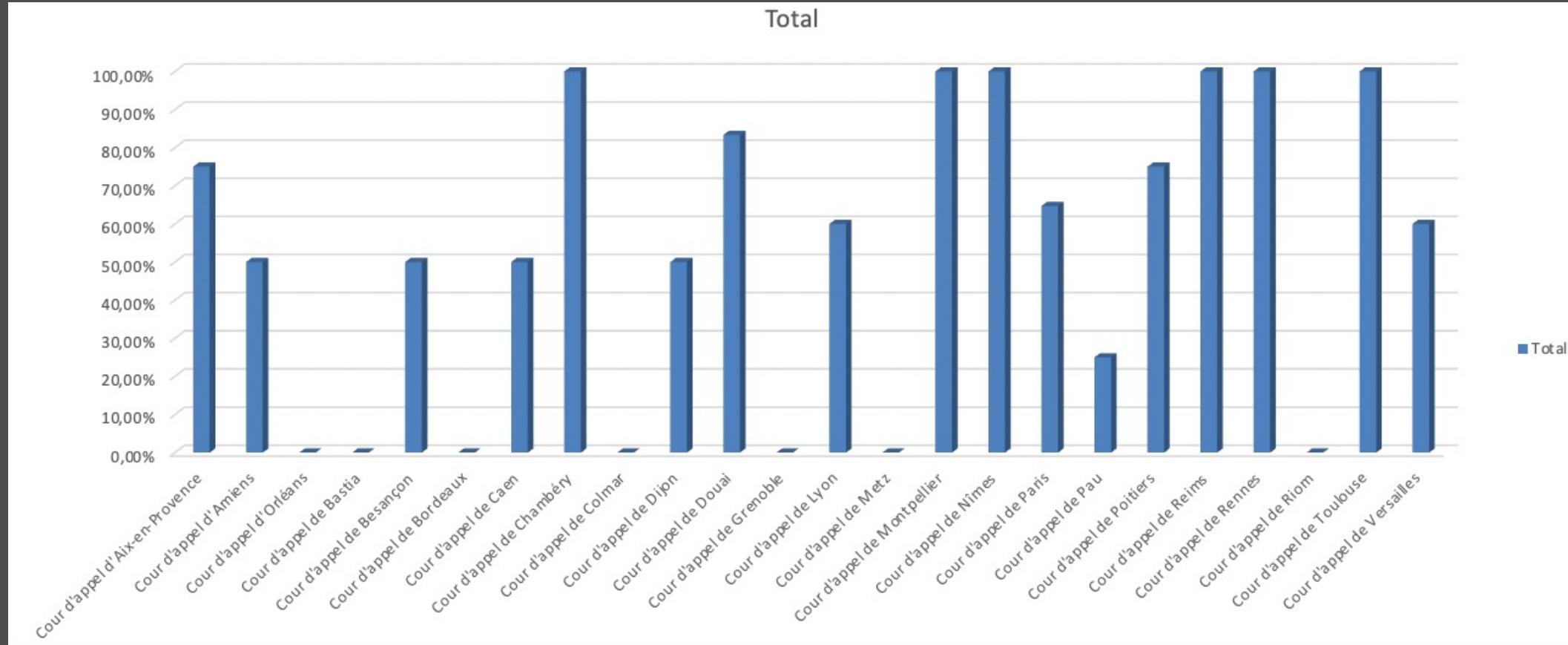
RATE OF CONFIRMATION BY COURT OF APPEAL (FROM 1998 TO 2023)



KEY POINTS

- Lack of homogeneity between decisions issued by the Courts of Appeal
- **Some Courts of Appeal almost systematically (>70%) confirm first instance decisions:** Bourges Court of Appeal, Nouméa Court of Appeal, Toulouse Court of Appeal
- **Some Courts of Appeal overturn most of the time (>50%) first instance decisions:** Angers Court of Appeal, Orléans Court of Appeal, Bordeaux Court of Appeal, Grenoble Court of Appeal
- Aim of this analysis : *Is it useful to form an appeal of the first instance's decision ?*

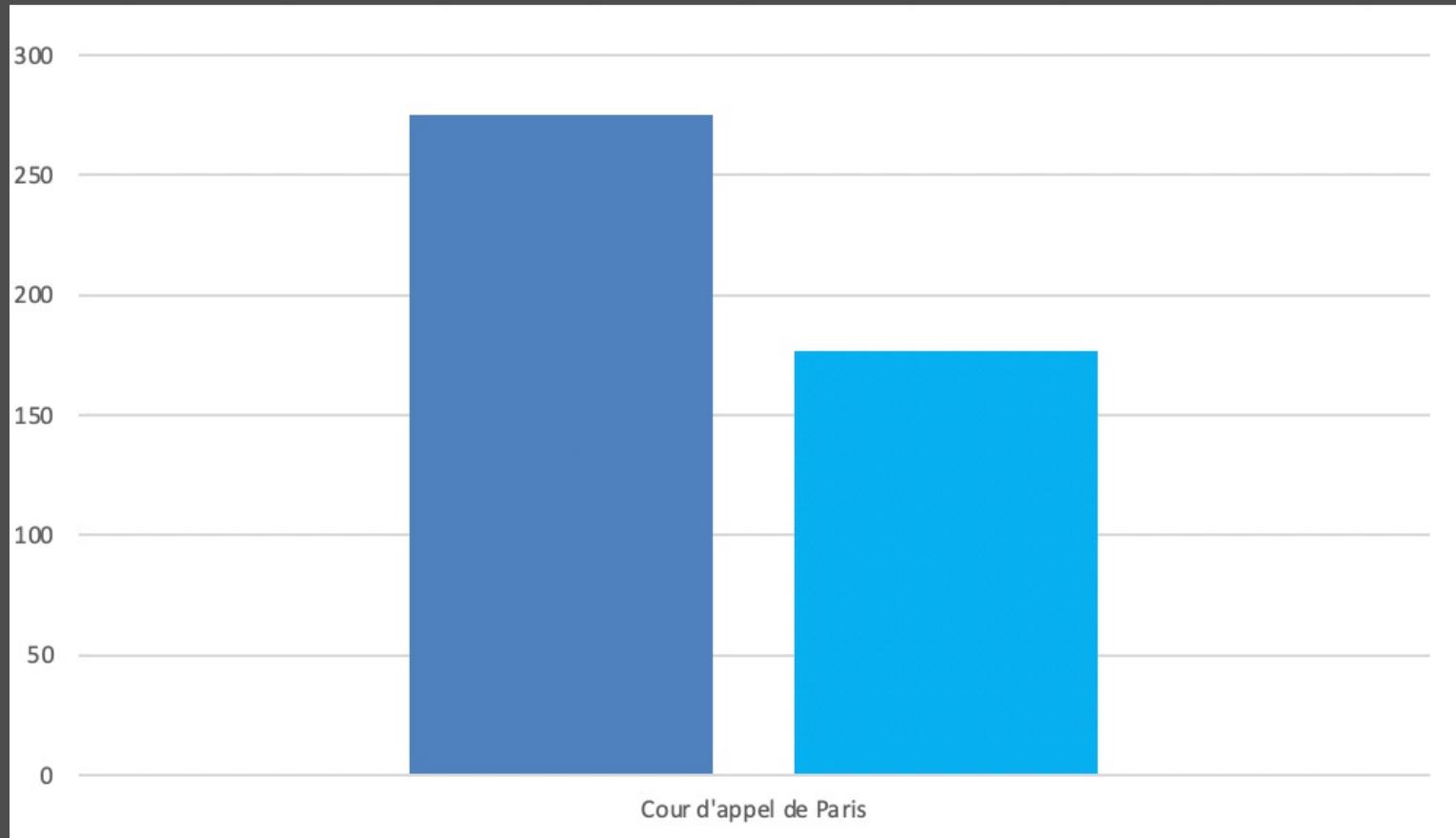
RATE OF CONFIRMATIONS BY COURT OF APPEAL (2023 ONLY)



KEY POINTS

- Most Courts of Appeal confirmed first instance decisions – with several Courts of Appeal confirming almost 100% of the time first instance decisions (Toulouse Court of Appeal, Montpellier Court of Appeal, Reims Court of Appeal)
- Except for several Courts of Appeal (Pau Court of Appeal, Dijon Court of Appeal, Amiens Court of Appeal)

FOCUS ON PARIS COURT OF APPEAL (FROM 1998 TO 2023)



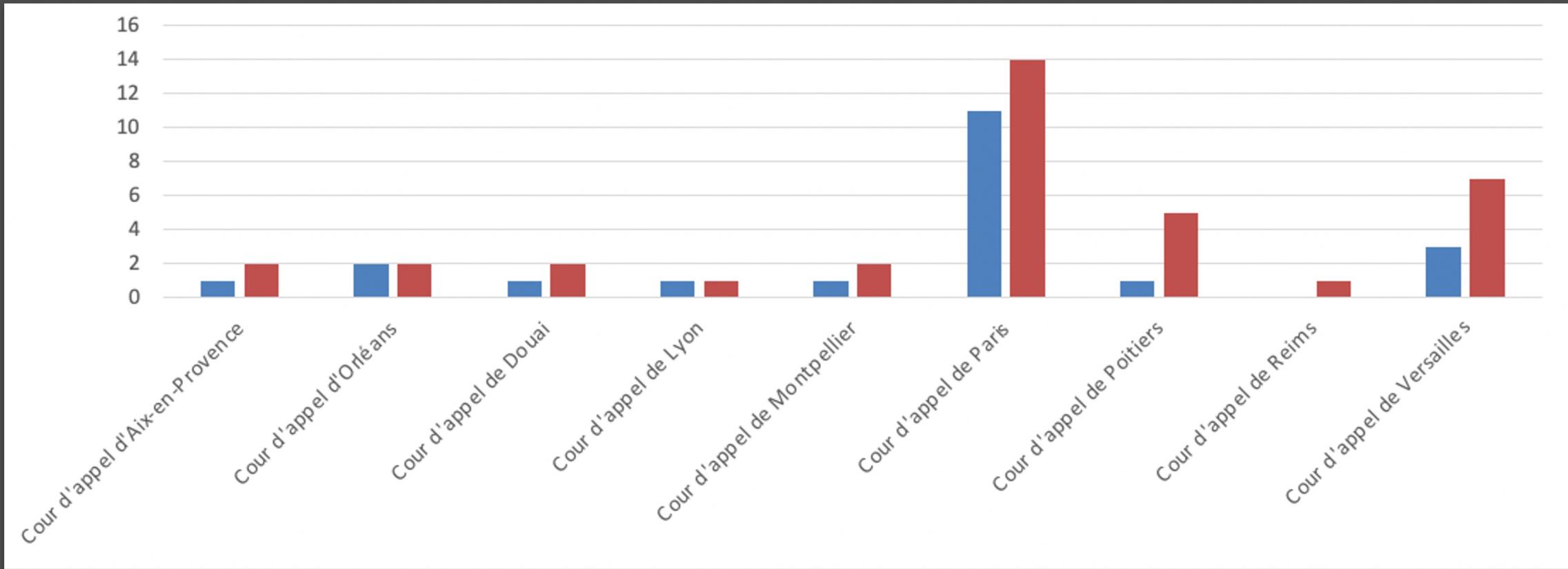
Total number of decisions

Number of decisions confirmed on appeal

KEY POINTS

- Total number of decisions: 275
➡ 28.8% of total decisions
- Number of decisions confirmed on appeal: 177
- Percentage of first instance decisions imposing insolvency law sanctions confirmed on appeal: 64,3%
- Average of first instance decisions imposing insolvency law sanctions confirmed on appeal by Paris Court of Appeal > Average of all Courts of Appeal
➡ 64,3% > 58,7%

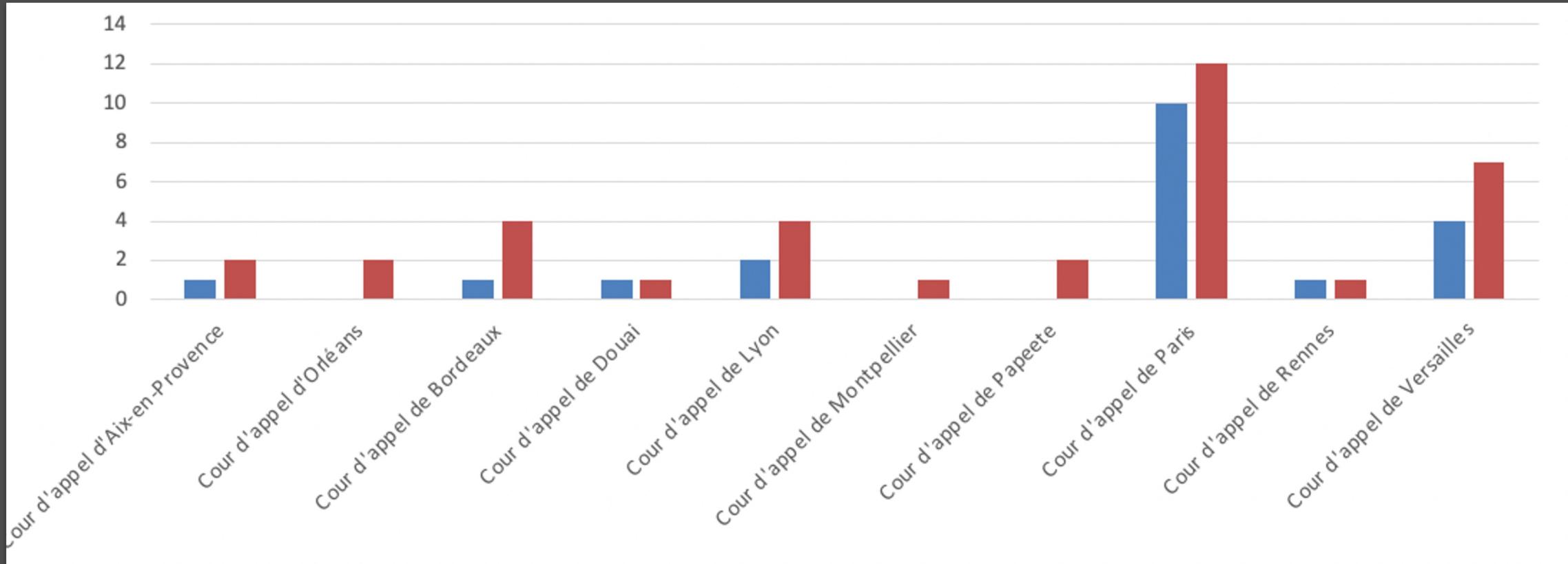
NUMBER OF DECISIONS IN 2020



KEY POINTS

- **Total number of decisions:** 36
- **Number of decision confirming:** 21
- **Percentage of first instance decisions imposing insolvency law sanctions confirmed on appeal:** 58,3%

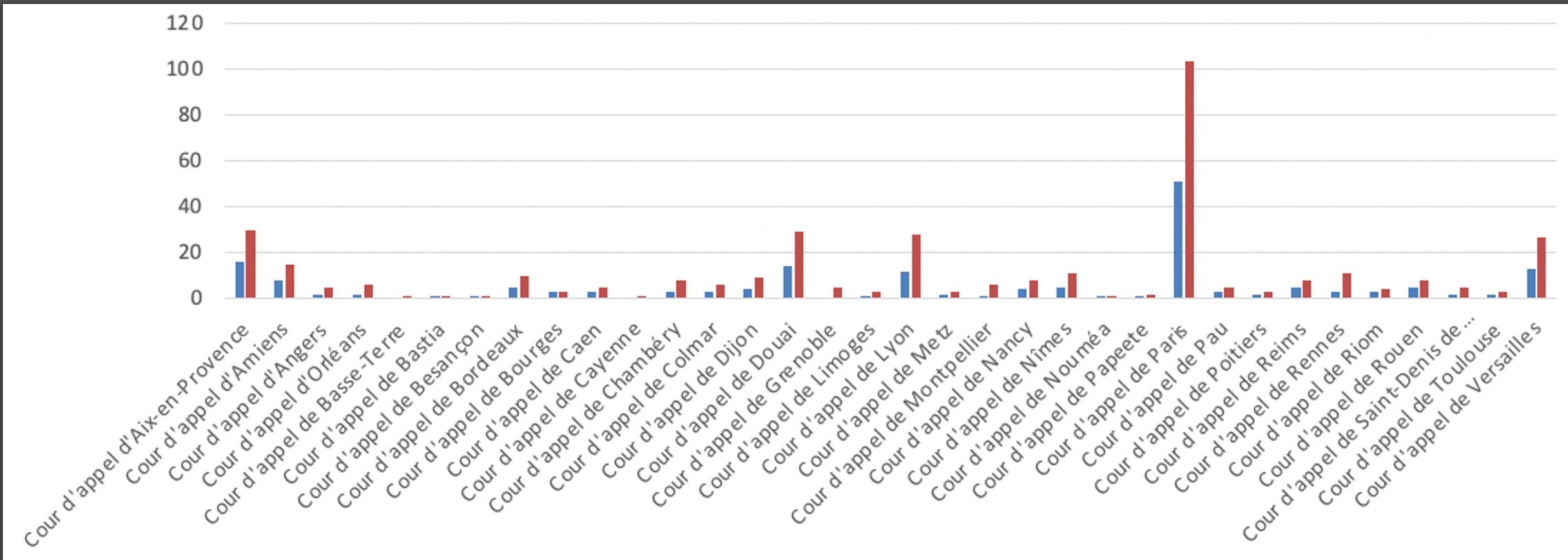
NUMBER OF DECISIONS IN 2021



KEY POINTS

- **Total number of decisions:** 36
- **Number of decision confirming:** 20
- **Percentage of first instance decisions imposing insolvency law sanctions confirmed on appeal:** 55,5 %

NUMBER OF DECISIONS IN 2022



KEY POINTS

- Total number of decisions: 375
- Number of decision confirming: 181
- Percentage of first instance decisions imposing insolvency law sanctions confirmed on appeal: 48,2%

STEP 4: CONCLUSION

CONCLUSION

- This analysis allows us to see where the most decisions are ruled.
- The French economy is focused in big cities like Paris or in the Parisian area (Versailles).
- Some Courts of appeal overrule the decision of the first instance Court more easily.
- Useful information to know when a client is convicted.
- *Did Covid-19 had an impact on the number of decisions ?*