Data set selection

My project is to predict the results of divisions (roll call votes) in Dáil Éireann based on the features of the preceding debate, using data available from the Oireachtas Open Data APIs. This project appealed to me because I have a personal interest in Irish politics, and have previously interned in Dáil Éireann for the current Ceann Comhairle, Deputy Seán Ó Fearghaíl, as part of a Master's programme in Public Affairs and Political Communication. As I have significant domain knowledge, I will be familiar with the concepts and terminology present in this data set and be able to focus on the machine learning aspects of the project.

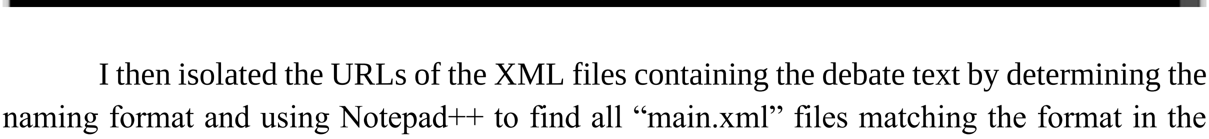
While there have been some prior studies of machine learning being applied to parliamentary questions in Ireland, there has been no in-depth study predicting results of votes based on either legislative text or debates. If successful, this study may provide new insights into parliamentary activity and many stakeholders would be interested in any techniques that could improve forecasting of future parliamentary votes. It would also be interesting to contrast any findings with those of studies performed in the US and UK, as Ireland's political landscape and system differ significantly from those countries.

The Oireachtas Open Data APIs consist of 9 interfaces which can be used to access various data sets, of which I will be mainly using the debates interface. My project will be primarily looking at data for divisions and debates. The data available on divisions includes the names of the deputies, their vote (yes/no/abstain), while the data available on debates includes information such as date, time, duration, number of distinct speakers, number of questions, the procedural structure of the debate, identities of the speakers, and full transcript.

Data set preparation

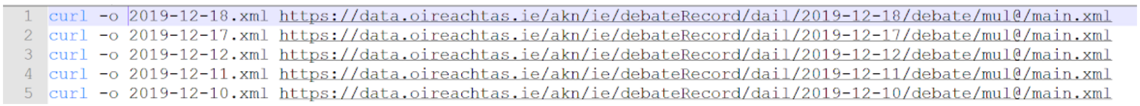
The Oireachtas Open Data portal provides an interface to retrieve filtered JSON data. This interface allows the user to select filters and generates a curl command based on the selection. The curl command retrieves a JSON file which contains summary data as well as links to XML documents with a structured representation of the full transcript of Oireachtas proceedings for a given day.

After running some initial tests using various filters to understand how they work and ensure I was downloading the correct data, I selected the dates of the 32nd Dáil and limited the query to only the Dáil (excluding the Seanad) to generate curl commands. I then ran these curl commands from the command prompt to download the JSON files of the debate data from the Oireachtas Open Data portal.



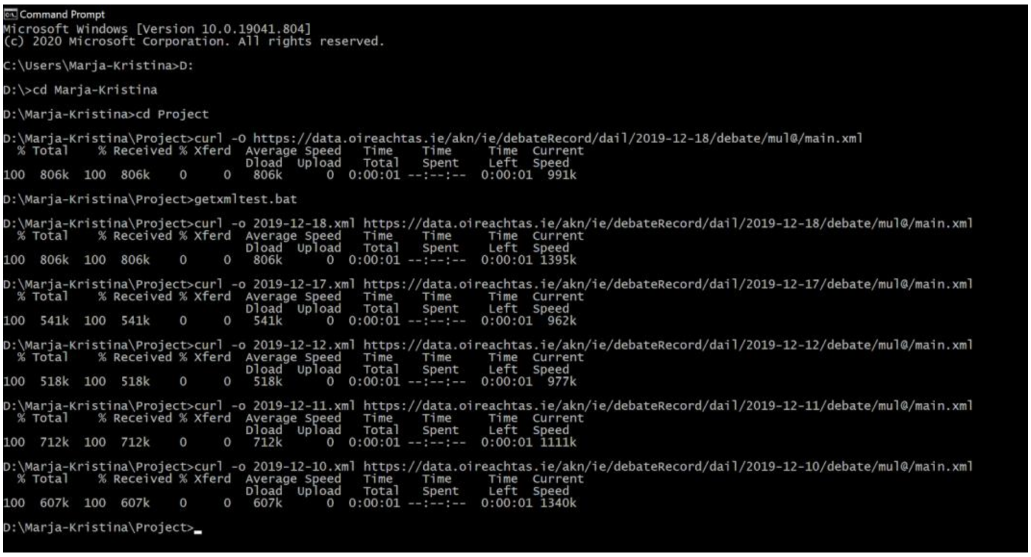
```
C:\Users\Marja-Kristina>curl -X GET "https://api.oireachtas.ie/v1/debates?chamber_type=house&chamber_id=&chamber=dail&da
te_start=2016-03-10&date_end=2020-01-14&limit=500" -H  "accept: application/json" > alldebates.json
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 10.4M  100 10.4M    0     0  2147k      0  0:00:05  0:00:05 --:--:-- 2299k
```

I then isolated the URLs of the XML files containing the debate text by determining the naming format and using Notepad++ to find all "main.xml" files matching the format in the debates JSON file. I then used these filenames to construct a batch file consisting of a separate curl command to download each day’s debate. As all the files had the same name on the server, the batch file also renamed them according to their date.



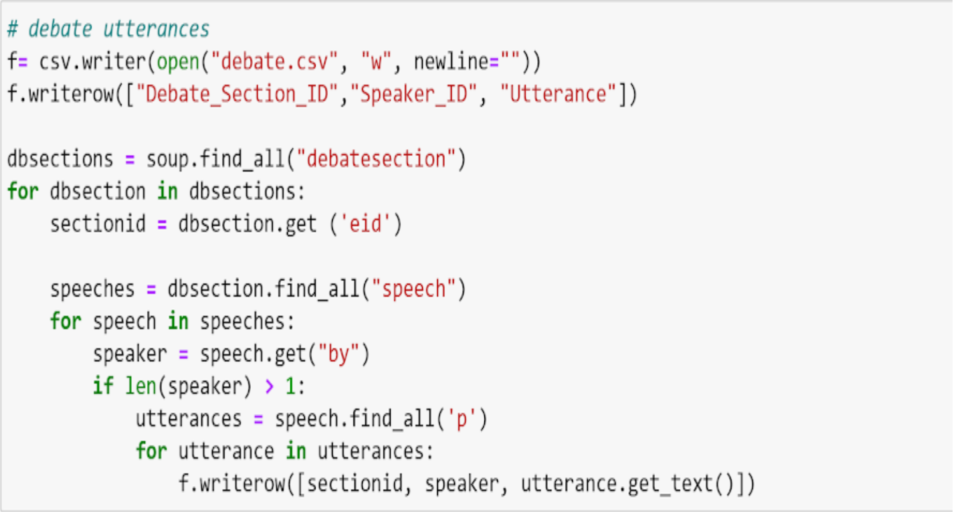
```
curl -o 2019-12-18.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-18/debate/mul@/main.xml
curl -o 2019-12-17.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-17/debate/mul@/main.xml
curl -o 2019-12-12.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-12/debate/mul@/main.xml
curl -o 2019-12-11.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-11/debate/mul@/main.xml
curl -o 2019-12-10.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-10/debate/mul@/main.xml
```

After testing a small sample, I ran the batch file which downloaded and renamed approximately 400 XML files each containing the full transcript of one day’s debate. This completed the retrieval of the raw data from the Oireachtas Open Data portal.



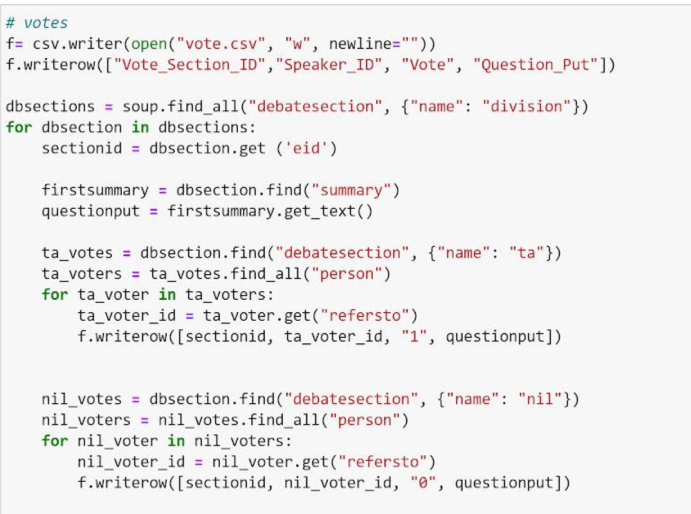
```
Microsoft Windows [Version 10.0.19041.804]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Marja-Kristina>D:

D:\>cd Marja-Kristina

D:\Marja-Kristina>cd Project

D:\Marja-Kristina\Project>curl -O https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-18/debate/mul@/main.xml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  806k  100  806k    0     0   806k      0  0:00:01 --:--:--  0:00:01  991k

D:\Marja-Kristina\Project>getxmltest.bat

D:\Marja-Kristina\Project>curl -o 2019-12-18.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-18/debate/mul@/main.xml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  806k  100  806k    0     0   806k      0  0:00:01 --:--:--  0:00:01 1395k

D:\Marja-Kristina\Project>curl -o 2019-12-17.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-17/debate/mul@/main.xml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  541k  100  541k    0     0   541k      0  0:00:01 --:--:--  0:00:01  962k

D:\Marja-Kristina\Project>curl -o 2019-12-12.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-12/debate/mul@/main.xml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  518k  100  518k    0     0   518k      0  0:00:01 --:--:--  0:00:01  977k

D:\Marja-Kristina\Project>curl -o 2019-12-11.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-11/debate/mul@/main.xml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  712k  100  712k    0     0   712k      0  0:00:01 --:--:--  0:00:01 1111k

D:\Marja-Kristina\Project>curl -o 2019-12-10.xml https://data.oireachtas.ie/akn/ie/debateRecord/dail/2019-12-10/debate/mul@/main.xml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  607k  100  607k    0     0   607k      0  0:00:01 --:--:--  0:00:01 1340k

D:\Marja-Kristina\Project>_
```

The next stage was to transform the data into a usable format. I used Jupyter Notebooks to load and manipulate the data and the BeautifulSoup library to filter for specific XML tags relevant to the task. I found the “prettify” function useful for understanding the XML structure. The first BeautifulSoup script I wrote isolated debate data from the XML, in particular the IDs of those TDs who spoke during the debate and the text of what they had said.



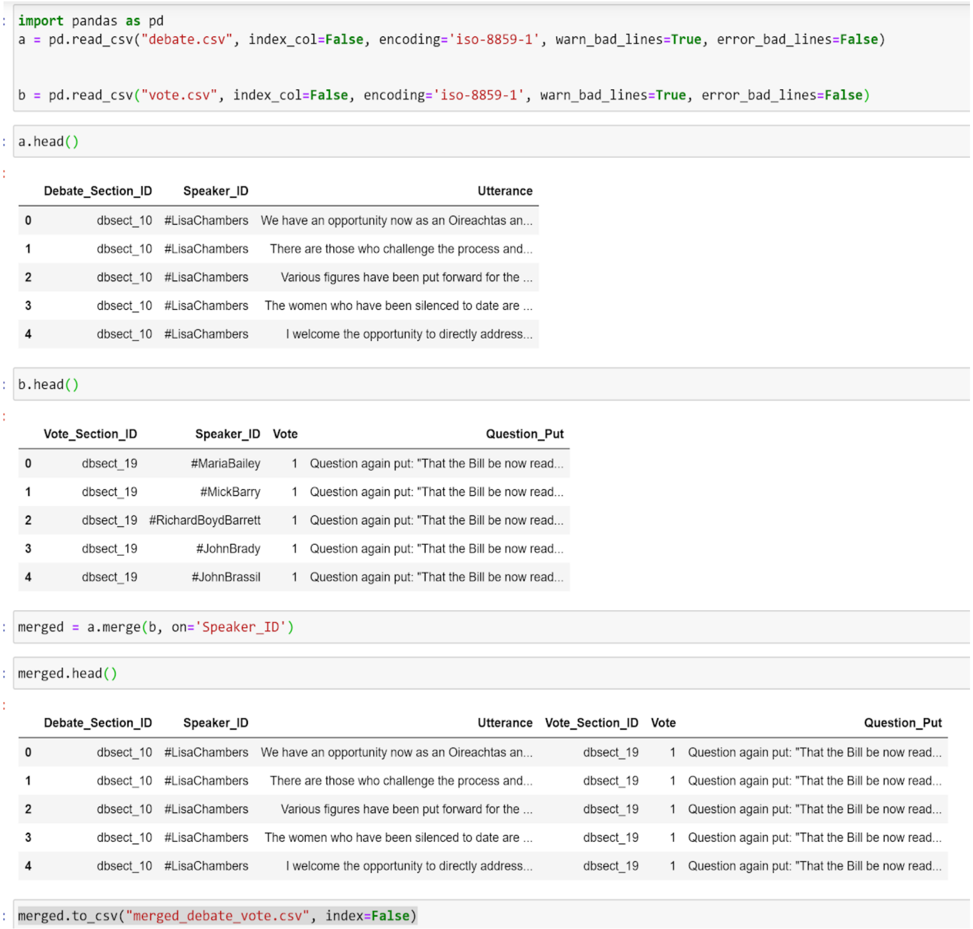
```
# debate utterances
f= csv.writer(open("debate.csv", "w", newline=""))
f.writerow(["Debate_Section_ID","Speaker_ID", "Utterance"])

dbsections = soup.find_all("debatesection")
for dbsection in dbsections:
    sectionid = dbsection.get ('eid')

    speeches = dbsection.find_all("speech")
    for speech in speeches:
        speaker = speech.get("by")
        if len(speaker) > 1:
            utterances = speech.find_all('p')
            for utterance in utterances:
                f.writerow([sectionid, speaker, utterance.get_text()])
```

A second script extracted the details of a particular vote including the IDs of the TDs who voted, and the type of vote they cast (‘Tá’/’Yes’ or ‘Níl’/’No’).



```python
# votes
f= csv.writer(open("vote.csv", "w", newline=""))
f.writerow(["Vote_Section_ID","Speaker_ID", "Vote", "Question_Put"])

dbsections = soup.find_all("debatesection", {"name": "division"})
for dbsection in dbsections:
    sectionid = dbsection.get ('eid')

    firstsummary = dbsection.find("summary")
    questionput = firstsummary.get_text()

    ta_votes = dbsection.find("debatesection", {"name": "ta"})
    ta_voters = ta_votes.find_all("person")
    for ta_voter in ta_voters:
        ta_voter_id = ta_voter.get("refersto")
        f.writerow([sectionid, ta_voter_id, "1", questionput])


    nil_votes = dbsection.find("debatesection", {"name": "nil"})
    nil_voters = nil_votes.find_all("person")
    for nil_voter in nil_voters:
        nil_voter_id = nil_voter.get("refersto")
        f.writerow([sectionid, nil_voter_id, "0", questionput])
```

I then used Pandas to firstly merge these two data sets based on the common TD identifier (*dail_vote_intermediate.csv*), and secondly to group all of the utterances by the same TD into the same field which is the approach used in ParlVote[1] (*dail_vote_condensed.csv*).



```python
import pandas as pd
a = pd.read_csv("debate.csv", index_col=False, encoding='iso-8859-1', warn_bad_lines=True, error_bad_lines=False)


b = pd.read_csv("vote.csv", index_col=False, encoding='iso-8859-1', warn_bad_lines=True, error_bad_lines=False)
```

```python
a.head()
```

	Debate_Section_ID	Speaker_ID	Utterance
0	dbsect_10	#LisaChambers	We have an opportunity now as an Oireachtas an...
1	dbsect_10	#LisaChambers	There are those who challenge the process and...
2	dbsect_10	#LisaChambers	Various figures have been put forward for the ...
3	dbsect_10	#LisaChambers	The women who have been silenced to date are ...
4	dbsect_10	#LisaChambers	I welcome the opportunity to directly address...

```python
b.head()
```

	Vote_Section_ID	Speaker_ID	Vote	Question_Put
0	dbsect_19	#MariaBailey	1	Question again put: "That the Bill be now read...
1	dbsect_19	#MickBarry	1	Question again put: "That the Bill be now read...
2	dbsect_19	#RichardBoydBarrett	1	Question again put: "That the Bill be now read...
3	dbsect_19	#JohnBrady	1	Question again put: "That the Bill be now read...
4	dbsect_19	#JohnBrassil	1	Question again put: "That the Bill be now read...

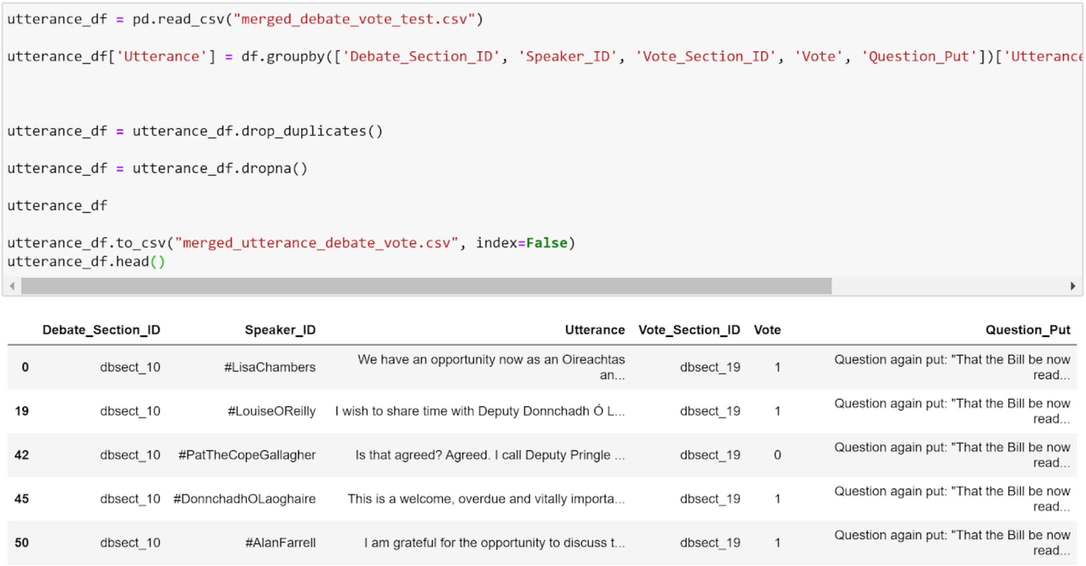
```python
merged = a.merge(b, on='Speaker_ID')
```

```python
merged.head()
```

	Debate_Section_ID	Speaker_ID	Utterance	Vote_Section_ID	Vote	Question_Put
0	dbsect_10	#LisaChambers	We have an opportunity now as an Oireachtas an...	dbsect_19	1	Question again put: "That the Bill be now read...
1	dbsect_10	#LisaChambers	There are those who challenge the process and...	dbsect_19	1	Question again put: "That the Bill be now read...
2	dbsect_10	#LisaChambers	Various figures have been put forward for the ...	dbsect_19	1	Question again put: "That the Bill be now read...
3	dbsect_10	#LisaChambers	The women who have been silenced to date are ...	dbsect_19	1	Question again put: "That the Bill be now read...
4	dbsect_10	#LisaChambers	I welcome the opportunity to directly address...	dbsect_19	1	Question again put: "That the Bill be now read...

```python
merged.to_csv("merged_debate_vote.csv", index=False)
```

[1] https://data.mendeley.com/datasets/czjfwgs9tm/2



```
utterance_df = pd.read_csv("merged_debate_vote_test.csv")

utterance_df['Utterance'] = df.groupby(['Debate_Section_ID', 'Speaker_ID', 'Vote_Section_ID', 'Vote', 'Question_Put'])['Utterance


utterance_df = utterance_df.drop_duplicates()

utterance_df = utterance_df.dropna()

utterance_df

utterance_df.to_csv("merged_utterance_debate_vote.csv", index=False)
utterance_df.head()
```

	Debate_Section_ID	Speaker_ID	Utterance	Vote_Section_ID	Vote	Question_Put
0	dbsect_10	#LisaChambers	We have an opportunity now as an Oireachtas an...	dbsect_19	1	Question again put: "That the Bill be now read...
19	dbsect_10	#LouiseOReilly	I wish to share time with Deputy Donnchadh Ó L...	dbsect_19	1	Question again put: "That the Bill be now read...
42	dbsect_10	#PatTheCopeGallagher	Is that agreed? Agreed. I call Deputy Pringle ...	dbsect_19	0	Question again put: "That the Bill be now read...
45	dbsect_10	#DonnchadhOLaoghaire	This is a welcome, overdue and vitally importa...	dbsect_19	1	Question again put: "That the Bill be now read...
50	dbsect_10	#AlanFarrell	I am grateful for the opportunity to discuss t...	dbsect_19	1	Question again put: "That the Bill be now read...

So far, I have completed this process for a single debate on the *Thirty-sixth Amendment of the Constitution Bill 2018: Second Stage*, the bill that would allow for the referendum to repeal the Eighth Amendment to the Irish Constitution. Now that I have the sequence of steps to transform the raw data into this format, these steps can be adjusted and repeated throughout the next phases of the project. In the next steps I will need to automate this processing for the entire data set. One challenge I will need to address is the fact that the debates and votes do not always happen on the same day, so a simple rule may be inadequate to determine which vote corresponds to which debate. I have noticed that there is often a bill ID that is attached to a particular bill and also to the corresponding vote, which could help to overcome this issue.

I am currently anticipating limiting the data set timeframe to the 32nd Dáil (10 March 2016 – 14 January 2020) in order to have a specific measurable period with the same set of legislators. This may need to be adjusted once it is clear whether there is sufficient training data in this timeframe. I will also need to assess whether to take further pre-processing steps that may be beneficial, such as eliminating procedural language, or adding additional metadata on the speakers, such as party or constituency (which are available via the Oireachtas APIs). Some other issues I may need to address are formatting (as Irish characters are not always correctly rendered in certain encoding formats) and mixing of Irish and English in the speeches, which occasionally takes place.