data_cleaning_checkpoint

March 1, 2024

1 Data Cleaning

Xander de la Bruere, Jakob Gertsch, Sam Layton, Matt Mella, Wilson Stoddard

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from hmmlearn import hmm
from sklearn.metrics import confusion_matrix
from sklearn.mixture import GaussianMixture
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer
import os
import nltk
from nltk.stem import WordNetLemmatizer

nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /home/wilso27/nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

[ ]:
```
True
```

1.0.1 Assignment Guidelines

For this assignment, you need to report your initial data exploration. Consider some of the following points as examples. Upload a Notebook with comments describing what you discovered. The emphasis here is not on perfect formatting, but on showing that you have done your due diligence in exploring the data and thinking through the consequences of what you see.

1. (DONE) Before doing any exploration, consider when and how you plan on holding out data for model evaluation. The gold standard is to have totally independent data you have never seen before tucked away so you can evaluate model performance at the end, but there can be many reasons this may not work. Explain your choice. If you do want to hold out the same data every time, consider fixing a random seed when you make the test-train split.
   - (DONE) Beware that a test set for time-series data is trickier to build than for independent data points. If you have training data just before and just after the test data point, the correlation between them means the test depends a lot on the training data

and hence is a bad test. At least ensure your test set comes after your training data, and ideally in a separate data-collection session.

2. (DONE) Print out a few dozen rows of the data. Is there anything you didn't expect to see? What opportunities for data cleaning and feature engineering may be important? Take care of these things.
3. Plot a few individual time series and do a similar check. Is there anything unbelievable you see?
4. (DONE) How much data is missing? Is the distribution of missing data likely different from the distribution of non-missing data? How might you do a meaningful imputation (if needed)? Are there variables that should be dropped? Implement some initial solutions.
5. Is there any hint that the data you have collected is differently distributed from the actual application of interest? If so, is there a strategy, such as reweighing samples, that might help? Use a histogram or KDE to visualize the distribution of key variables. Consider log-scaling or other scaling of the axes. How should you think about outliers? Is there a natural scaling for certain variables?
6. Use 2D and/or 3D plot scatter plots, histograms, or heat maps to look for important relationships between variables. Consider using significance tests, linear model fits, or correlation matrices to clarify relationships.
7. Does what you see change any of your ideas for what models might be appropriate? Among other things, if your models rely on specific assumptions, is there a way you can check if these assumptions actually hold by looking at the data? If you are using linear models, do the relevant plots look linear? Is there some other scaling where the model assumptions might more nearly hold?

1.0.2 1. Data for Model Evaluation

Out dataset is naturally split into episodes. To evaluate our model, we will leave out a certain percentage of episodes to train the model, then test our model on entire episodes to see how successfully we can identify different speakers. Training on the same episode would fail to provide independent enough data, giving us too much information with the sentence before and after a test sentence.

However, by using separate episodes, we are testing on data that comes from a separate data collection session. This essentially matches the indicated gold standard of data, where we will permanently set aside a test set of episodes to train on. The test data is structured the same way as the training data, which will be ideal for evaluating our models on.

1.0.3 2. Basic Description of Data

Our dataset comes from consists of over 140 thousand radio transcripts from NPR. The episodes come from a span of 20 years between January 1999 and October 2019. The transcribed text represents over 10 thousand hours of audio. The data is stored in a CSV file and is summarized by the table below.

Column	Data Type	Description
`episode`	int	The episode number

Column	Data Type	Description
`episode_order`	int	The line number within each episode. Note that when a different person begins speaking, the row ends and another begins
`speaker`	str	The speaker and (usually) their title
`utterance`	str	A block of transcribed audio

```
[ ]: # read in data
path = os.getcwd()
df = pd.read_csv(f'{path}/archive/utterances.csv')

# remove all speakers labeled as _NO_SPEAKER
df = df[df['speaker'] != '_NO_SPEAKER']
display(df.head(10))
```

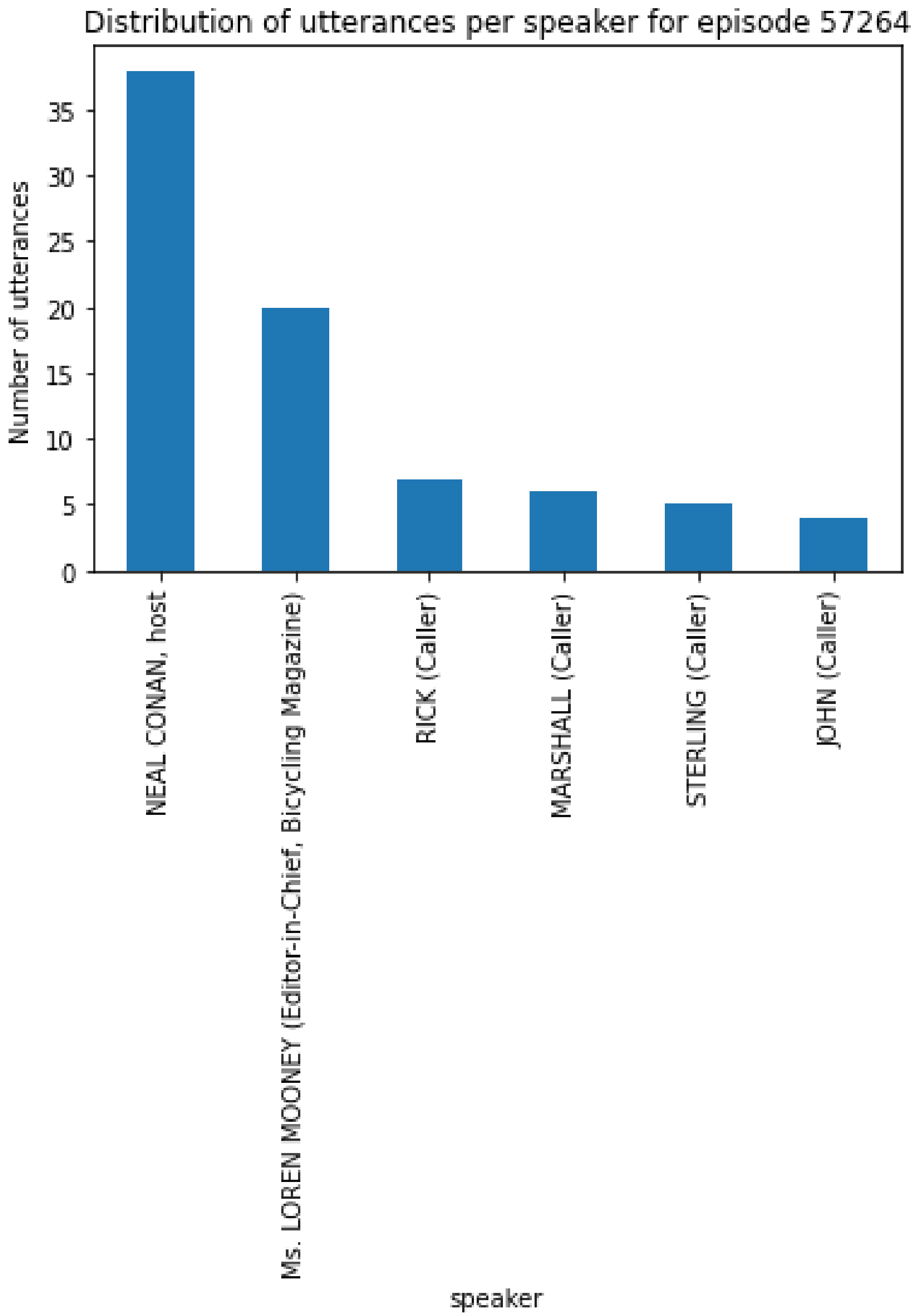
```
   episode  episode_order                                            speaker  \
0    57264              9  Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
1    57264             10  Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
2    57264             11                                   NEAL CONAN, host
3    57264             12  Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
4    57264             13                                   NEAL CONAN, host
5    57264             14  Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
6    57264             15                                   NEAL CONAN, host
7    57264             16  Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
8    57264             17                                   NEAL CONAN, host
9    57264             18                                      JOHN (Caller)

                                           utterance
0  It's a 2,200-mile race. To give some sense of …
1  So for a top competitor like Lance to try to m…
2  So in every team, presumably there's one star,…
3  That's right. Each team has nine riders. And w…
4  So slipstream, this is like drafting in car ra…
5                             That's exactly right.
6  And so the guy who's in back has an easier tim…
7  That's right. There's a lot of deal making tha…
8  We're talking with Loren Mooney, the editor-in…
9                                             Hello.
```

We aren't concerned with audio where there is no defined speaker (for example, the intro and outro), so we dropped these lines. In addition, we didn't expect to see any rows with `NaN` values in `utterance` so we dropped those, which is done in a later section. Other than the relatively few rows with `NaN` values, the data set was quite clean. Below we add one column, `word_count`, to keep track of how many words are in row of `utterance`.

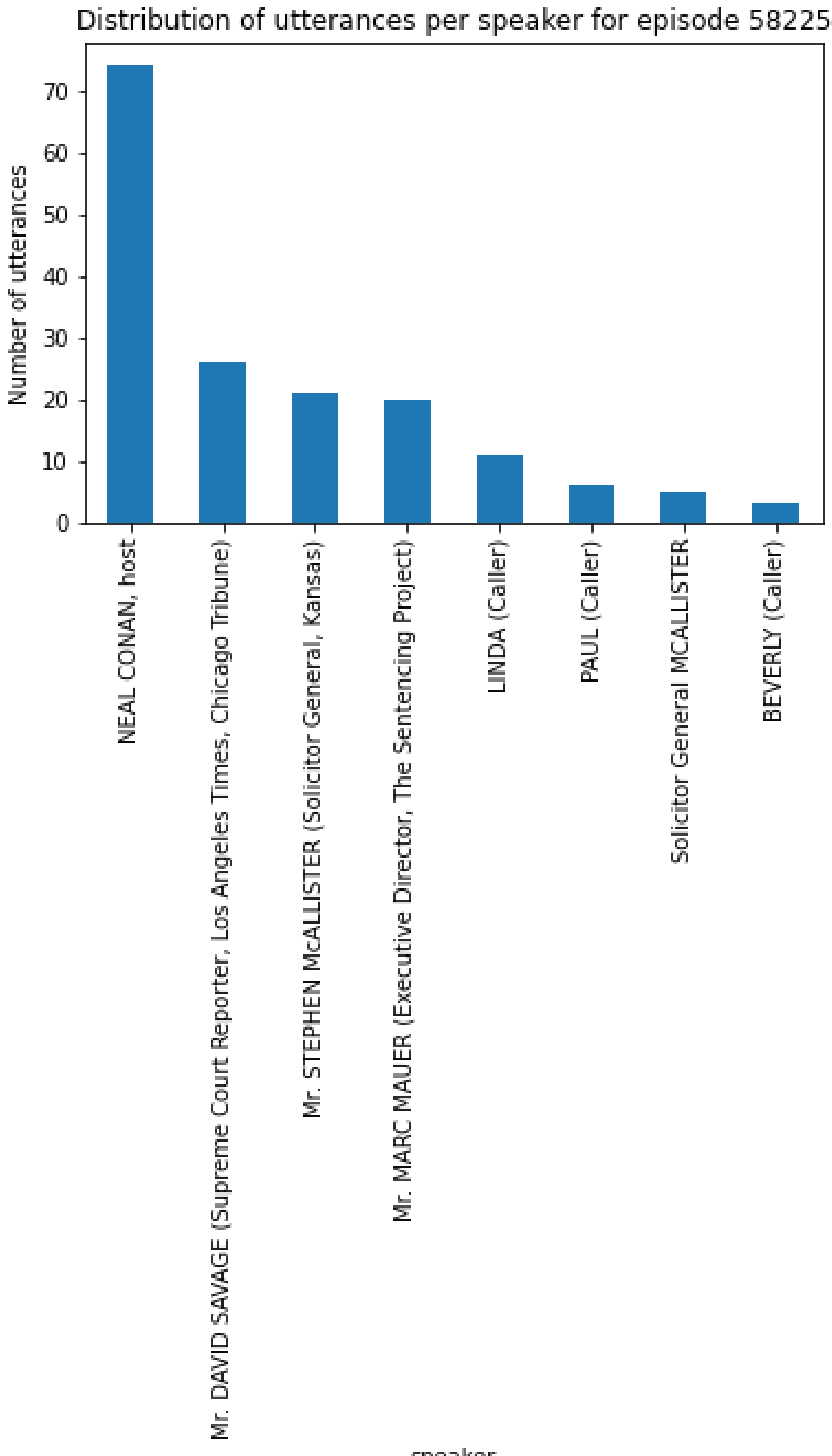
1.0.4 3. Individual Time Series Checks

[ ]:
```python
def plot_speaker_distribution_for_episode(df, episode):

    # print the episode
    print('Episode:', episode)

    # Filter the dataframe for the specific episode
    df_episode = df[df['episode'] == episode]

    # print the amount of speakers
    print('Speakers:', len(df_episode['speaker'].unique()))

    # Group by 'speaker' and count the number of rows
    speaker_counts = df_episode['speaker'].value_counts()

    # Plot a bar chart
    speaker_counts.plot(kind='bar')
    plt.ylabel('Number of utterances')
    plt.title(f'Distribution of utterances per speaker for episode {episode}')
    plt.show()

# iterate through the first 10 episodes
for episode in df['episode'].unique()[:4]:
    plot_speaker_distribution_for_episode(df, episode)
```

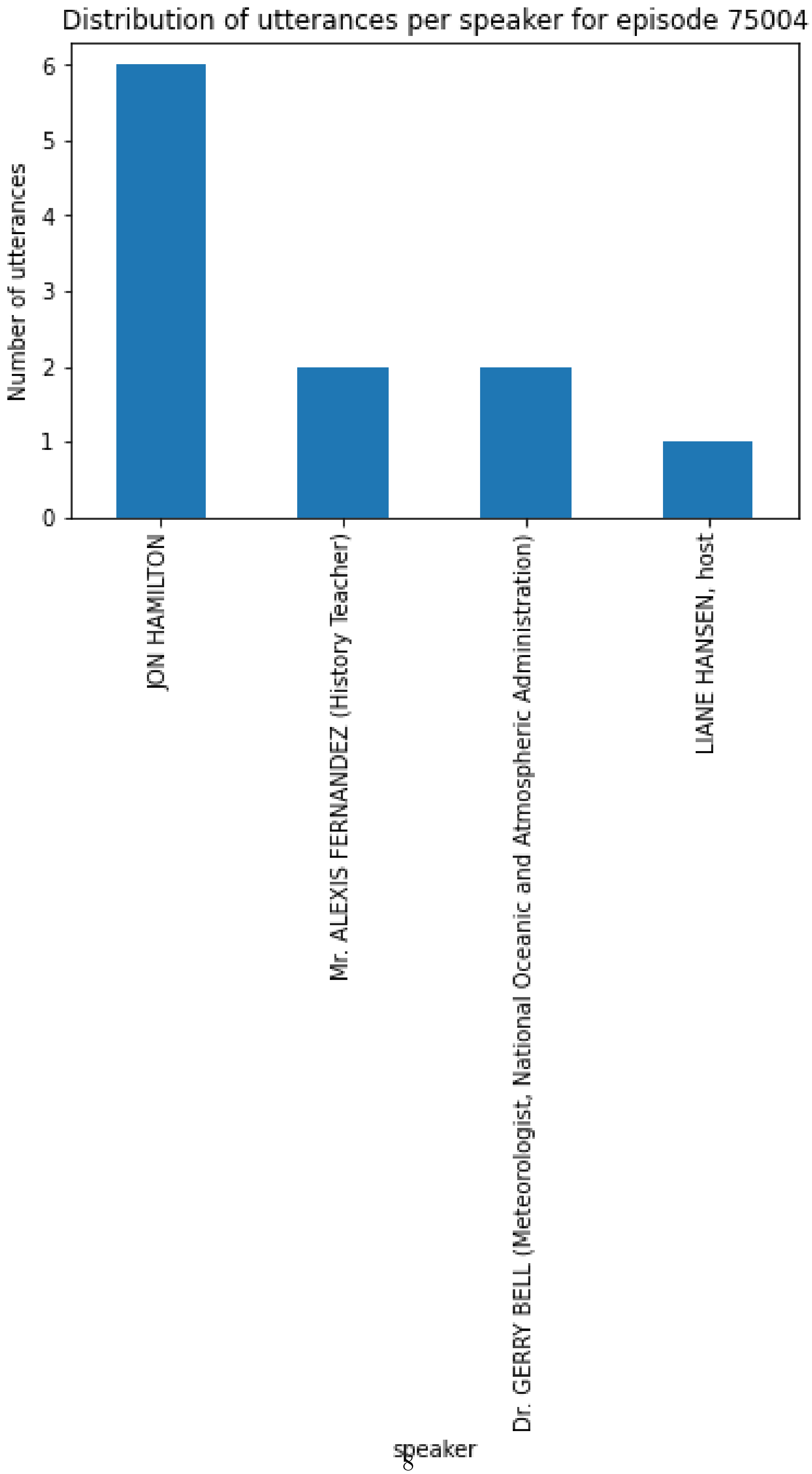
```
Episode: 57264
Speakers: 6
```




```
Episode: 58225
Speakers: 8
```

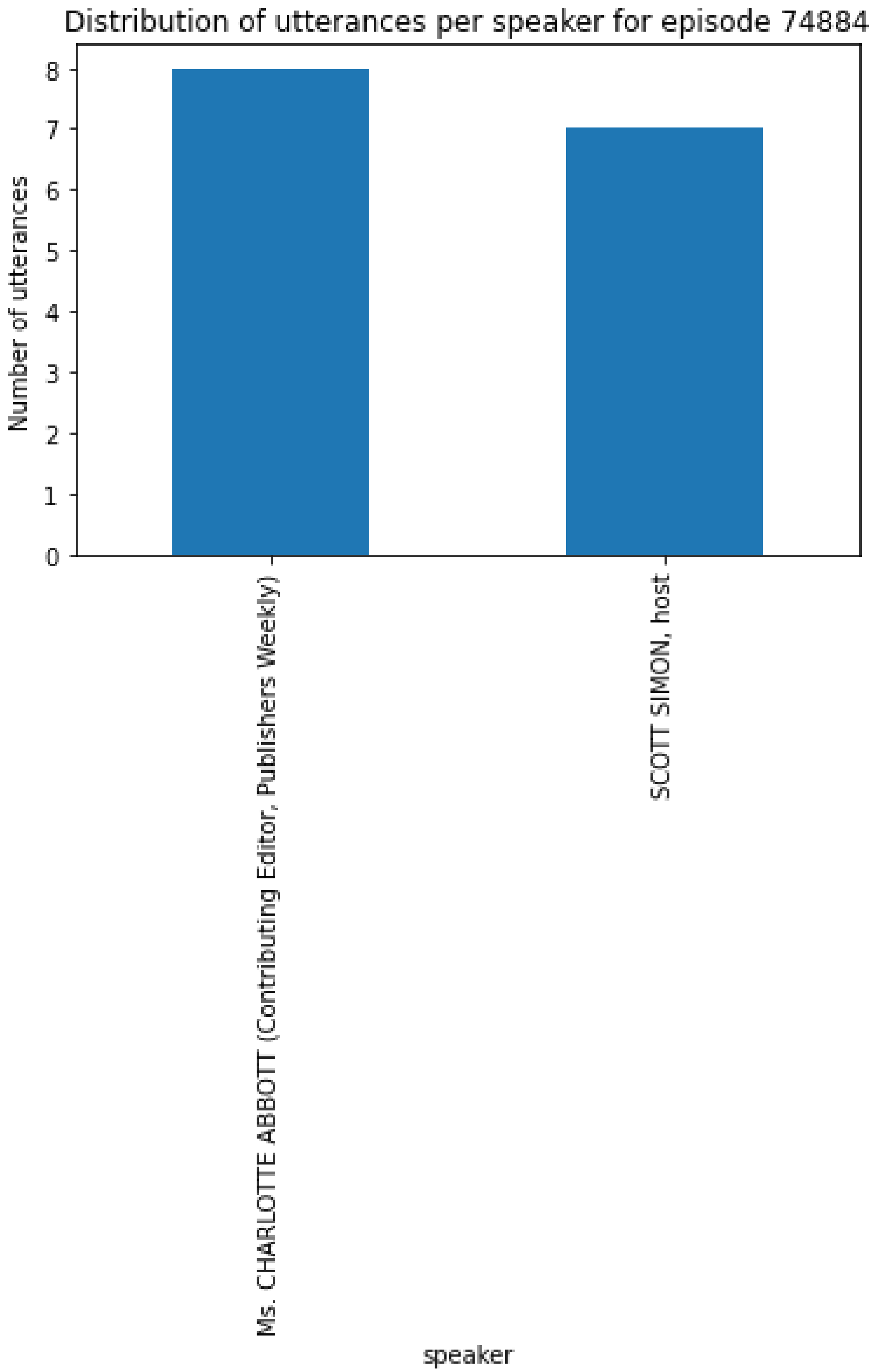

Distribution of utterances per speaker for episode 58225
Number of utterances
0
10
20
30
40
50
60
70
NEAL CONAN, host
Mr. DAVID SAVAGE (Supreme Court Reporter, Los Angeles Times, Chicago Tribune)
Mr. STEPHEN McALLISTER (Solicitor General, Kansas)
Mr. MARC MAUER (Executive Director, The Sentencing Project)
LINDA (Caller)
PAUL (Caller)
Solicitor General MCALLISTER
BEVERLY (Caller)
speaker

Episode: 75004
Speakers: 4



Distribution of utterances per speaker for episode 75004
Number of utterances
0
1
2
3
4
5
6
JON HAMILTON
Mr. ALEXIS FERNANDEZ (History Teacher)
Dr. GERRY BELL (Meteorologist, National Oceanic and Atmospheric Administration)
LIANE HANSEN, host
speaker

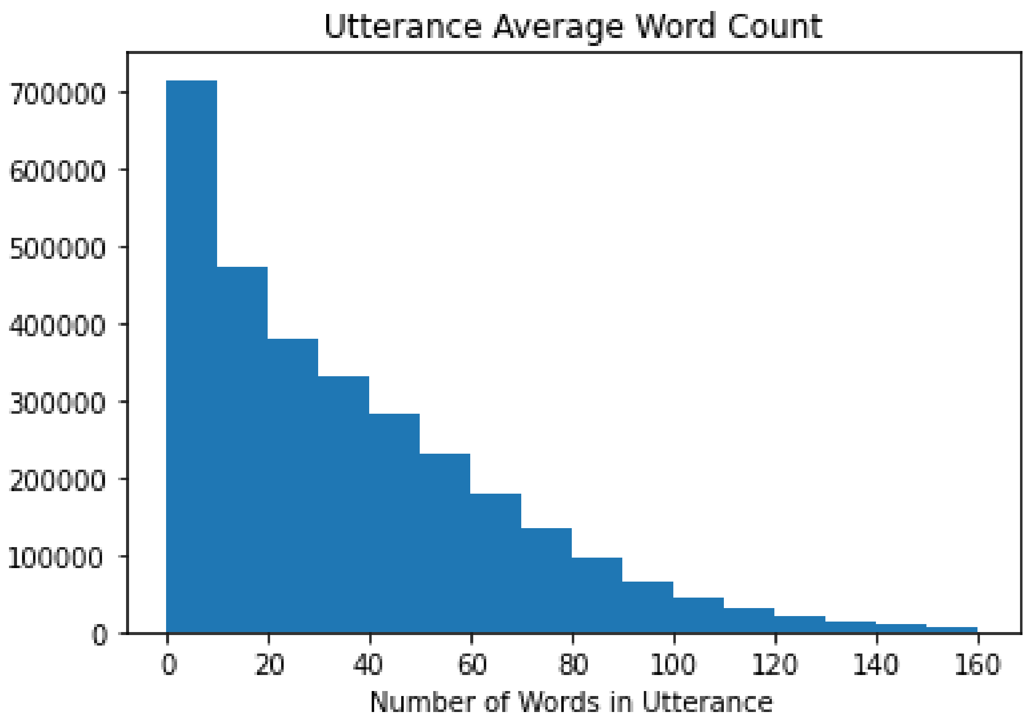
Episode: 74884
Speakers: 2



We will find that the distribution of speakers is not equally split. Because many of these conversations are interviews, we will find one speaker overwhelmingly dominates the conversation. Further, outside callers into the show will imply that the members of a conversation vary as callers join and leave the conversation.

OK here's a distribution of number of words per utterance!!!

[ ]:
```
# Redo the index and ensure all utterances are strings, then calculate the word␣
↪count
df = df.reset_index()
df['utterance'] = df['utterance'].astype(str)
df['word_count']=np.array([len(df['utterance'][i].split()) for i in␣
↪range(len(df))])

# Plot a histogram showing the relative distribution of word counts across␣
↪utterances
plt.hist(df['word_count'],bins=np.arange(0,170,10))
plt.xlabel("Number of Words in Utterance")
plt.title("Utterance Average Word Count")
plt.show()
```



[ ]:
```
df2sp = pd.read_csv('archive/utterances-2sp.csv')
```

[ ]:
```
display(df2sp.head(10))
```

```
   episode  episode_order  turn_order  speaker_order  host_id  is_host  \
0        1              1           0              0        0     True
1        1              1           1              0        0     True
2        1              1           2              0        0     True
3        1              1           3              0        0     True
4        1              2           0              0        0     True
5        1              3           0              1       -1    False
6        1              4           0              0        0     True
7        1              4           1              0        0     True
8        1              5           0              1       -1    False
9        1              5           1              1       -1    False

                                           utterance
0  The impeachment inquiry picks up tomorrow wher…
1  Just this morning, the lawyer for the whistleb…
2                     There's are a lot of moving parts.
3   Fortunately, NPR's Mara Liasson is here to help.
4                                      Good morning.
5                                Good morning, Lulu.
6                                         All right.
7                                 What's the latest?
8  Well, the latest is that the lawyer for the fi…
9  The first whistleblower only had second and th…
```

1.0.5 This is an alternative dataset of just 2-person conversations. This dataset limits what we are modeling to just 2 different speakers, but will avoid the issue of variable participants in a conversation

[ ]:
```python
# remove all speakers labeled as _NO_SPEAKER
df = df[df['speaker'] != '_NO_SPEAKER']
print(len(df))

# group by episode
episodes = df.groupby('episode')

# show the first episode, 57264
display(episodes.get_group(57264))
```

```
3015435

         level_0    index  episode  episode_order  \
0              0        0    57264              9
1              1        1    57264             10
2              2        2    57264             11
3              3        3    57264             12
4              4        4    57264             13
…            …        …       …               …
3015087  3015087  3199423    57264              4
```

```
3015088  3015088  3199424   57264            5
3015089  3015089  3199425   57264            6
3015090  3015090  3199426   57264            7
3015091  3015091  3199427   57264            8

                                                          speaker  \
0         Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
1         Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
2                                        NEAL CONAN, host
3         Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
4                                        NEAL CONAN, host
…                                                      …
3015087                                  NEAL CONAN, host
3015088   Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
3015089                                  NEAL CONAN, host
3015090   Ms. LOREN MOONEY (Editor-in-Chief, Bicycling M…
3015091                                  NEAL CONAN, host

                                                        utterance  word_count
0         It's a 2,200-mile race. To give some sense of …          50
1         So for a top competitor like Lance to try to m…          87
2         So in every team, presumably there's one star,…          33
3         That's right. Each team has nine riders. And w…         118
4         So slipstream, this is like drafting in car ra…          10
…                                                      …          …
3015087   Joining us now from our bureau in New York is …          33
3015088                               Thanks for having me.           4
3015089   And I've got my copy of Bicycling Magazine, an…          23
3015090   Well, yes, it's true. Actually, the race did b…          44
3015091   Well, this is a race that lasts 2,000 miles. T…          16

[80 rows x 7 columns]
```

1.0.6 4. Missing Data

As seen below, our dataframe has relatively few rows with `NaN` value, all of which appear in the `utterance` column. We drop these rows without any material effect to our analysis. Again, since we are concerned with identifying speakers, we can safely drop entries where nothing is said. No imputations are necessary to handle missing values as we simply drop them.

[ ]:
```python
# Check for missing values
print(df.isnull().sum())

# Check for nonstring values in the 'utterance' column
print(df['utterance'].apply(type).value_counts())

# View 10 of the nonstring values in the 'utterance' column in the original␣
↪dataframe
```

```
display(df[df['utterance'].apply(type) != str].head(10))

# remove NaN values
df = df.dropna()

# check for missing values
print(df.isnull().sum())
```

```
level_0         0
index           0
episode         0
episode_order   0
speaker         0
utterance       0
word_count      0
dtype: int64
utterance
<class 'str'>    3015435
Name: count, dtype: int64
```

```
Empty DataFrame
Columns: [level_0, index, episode, episode_order, speaker, utterance, word_count]
Index: []
```

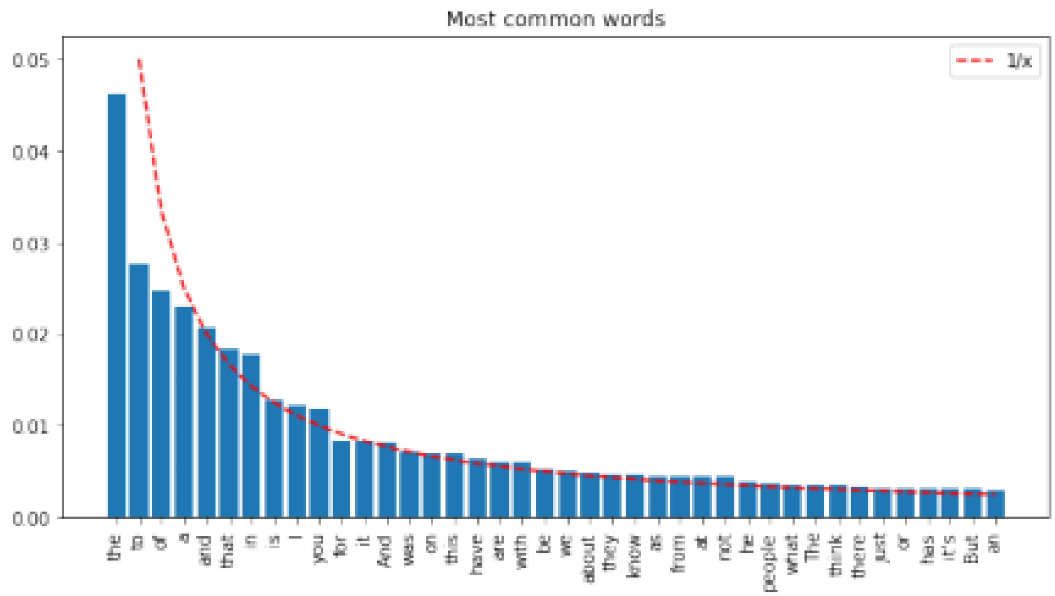
```
level_0         0
index           0
episode         0
episode_order   0
speaker         0
utterance       0
word_count      0
dtype: int64
```

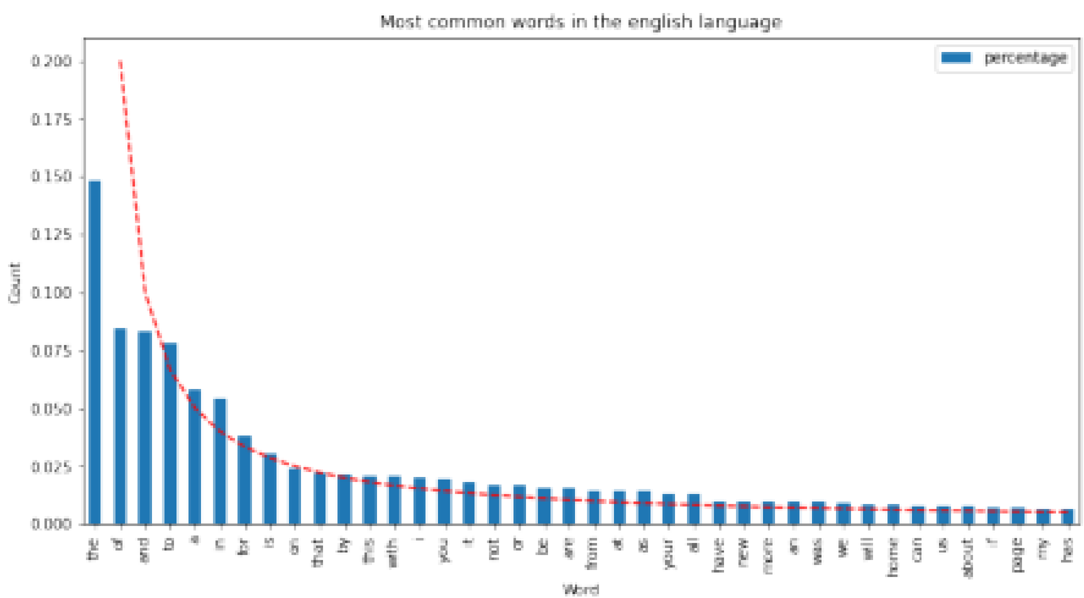
1.0.7 5. Distribution of Data

The theoretical observation space is the space of all conversations. That is a very large space to work with. We are retrieving data mainly from NPR thus the data is expected to be skewed in some ways. Podcasts tend to be more formal. Small talk is not likely to be represented since these are interview-like conversations. That being said, the main structure of conversations will likely be preserved in the data. Below is a comparison of our dataset with the Google Web Trillion Word Corpus that pulls data from every text available to Google which we can assume is a good sampling of what the true distribution of english would approach.

[ ]:
```
plt.figure(figsize=(10,10))
plt.subplot(211)
plt.imshow(plt.imread('most_common_dataset.png'))
plt.axis('off')

plt.subplot(212)
```

```python
plt.imshow(plt.imread('most_common_english.png'))
plt.axis('off')
plt.show()
```



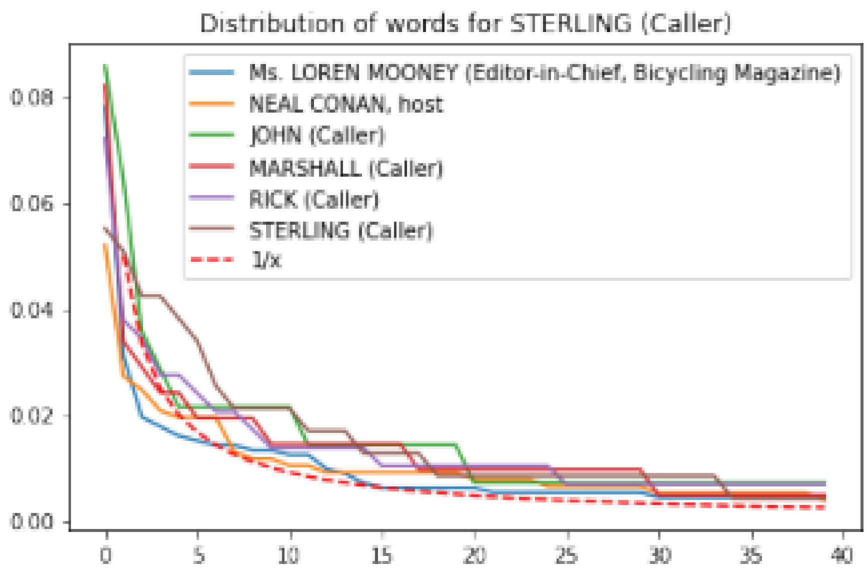


The first image is of our dataset and the second image is of the Google Web Trillion Word Corpus. We see there is a larger percentage of stopping words like "the" and "of" which is to be expected as the text becomes more diverse. The top 40 words in each dataset are mostly similar with the same general distribution. This gives us confidence that the NPR dataset will serve our purposes as a labeled conversation dataset.

1.0.8 6. Visualizations of Data Relationships

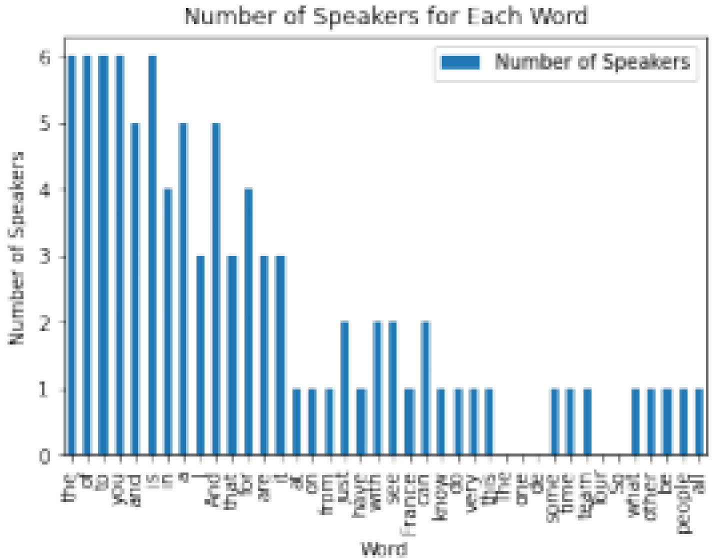
Here are some bar plots of word distributions by speaker for one episode of NPR. The first plot is the shape of the distribution of top 40 words by speaker. We see that the general distributions most common words is similar across the speakers.

[ ]:
```python
# Load and plot the image
image = plt.imread('comp_of_speaker_dist.png')
plt.imshow(image)
plt.axis('off')
plt.show()
```



However when we analyze which words are found in each speakers top 40 words we have a more relevant analysis. Here is a plot of the top 40 words overall on the x-axis with the number of individual speakers (out of 6 for the episode) that have that word in their individual top words. The sharp drop off shows the unique vocabularies found among these 6 speakers even when talking about the same topic.

[ ]:
```python
# Load and plot the image
image = plt.imread('number_speakers_by_word.png')
plt.imshow(image)
plt.axis('off')
plt.show()
```



This is very encouraging for us since our assumption is that by word usage we will be able to differentiate between speakers. It is further encouraging since this in no way accounts for sequential information like how those words are used in each speakers sentence structure.

We also lemmatized and vectorized the dataset to allow us to make other visualizations. Here is an example of what we found. This data represents two of the top hosts. The data is taking the vectorized rows in the dataset specific to these two speakers and plotted in two dimensions using PCA. We see clear differences being detected even if there is some overlap. The ovelap may cause some concern, however since they are speaking the same language this is to be expected. We should be able to work with the differences we do see.

[ ]:
```
# Load and plot the image
image = plt.imread('two_top_speakers_scatter.png')
plt.imshow(image)
plt.axis('off')
plt.show()
```

1.0.9 7. Inspiration Moving Forward

Our analysis reveals discernible differences in the data when categorized by speaker, suggesting that speaker diarization using text transcription could be a viable approach. However, we must proceed with caution as the data isn't distinctly separated and exhibits considerable overlap. Despite this, given that we haven't yet incorporated any sequential information provided by the Hidden Markov Model (HMM), we remain optimistic about achieving our objectives.

Furthermore, our observations support the hypothesis that unique speakers can be successfully identified through their specific vocabulary and speech patterns.