

Data Cleaning

```
In [56]: 1 from pickle import loads, dumps, load
2 import json
3 import re
4 import pandas as pd
5 from string import punctuation
6 import nltk
7 from nltk.stem.porter import PorterStemmer
8 from nltk.stem.wordnet import WordNetLemmatizer
9 #nltk.download('stopwords')
10 from nltk.corpus import stopwords
11 from pathlib import Path
12 import shutil
13 import os
14 from numpy import logical_and
15 from autocorrect import Speller
```

```
In [57]: 1 # Load the filepaths
2 with open('../raw_data/corpus_dict.pkl', 'rb') as handle:
3     corpus_dict = load(handle)
```

```
In [58]: 1 def load_test_df(season, episode):
2     raw_data_folder = "../raw_data/"
3     # Example df
4     with open(raw_data_folder+corpus_dict[season][episode], 'rb') as handle:
5         unserialized_data = load(handle)
6     test_df = loads(dumps(unserialized_data))
7     return test_df
```

Steps

- removing '
- remove quotes
- To lowercase
- remove HTML tags
- woman = female
- leerzeichen namen
- Delete empty tags / rows
- vor/nachname
- spell checking

source: <https://medium.com/analytics-vidhya/automated-keyword-extraction-from-articles-using-nlp-bfd864f41b34> (<https://medium.com/analytics-vidhya/automated-keyword-extraction-from-articles-using-nlp-bfd864f41b34>)

Further

Some characters are not characters. They are screen directions. They will all be labelled "interlude"

Preperation

```
In [59]: 1 # Odd character found in othr steps of the project
2 with open('characters_occurring_once.pkl', 'rb') as f1, open('long_chatacter_names.pkl', 'rb') as f2, open('odd_characters.pkl', 'rb') as f3:
3     characters_occurring_once = load(f1)
4     long_chatacter_names = load(f2)
5     odd_characters = load(f3)
6 odd_characters.remove('child 1')
7 odd_characters.remove('child 2')
8 odd_characters.remove('child')
```

```
In [60]: 1 def clean_text(string:str):
2     speller = Speller(lang='en')
3     text = string.lower()
4     text = text.strip()
5     text = re.sub('<[a-zA-A/]*>', '', text)
6     text = re.sub('\'', '"', text)
7     text = re.sub('"re', ' are', text)
8     text = re.sub('"t', ' not', text)
9     text = re.sub("I'm", 'I am', text)
10    text = speller(text)
11    text = re.sub('"', ' ', text)
12    #source: https://stackoverflow.com/questions/265960/best-way-to-strip-punctuation-from-a-string
13    text = text.translate(str.maketrans('', '', punctuation))
14    text = text.split()
15    text = [speller(word) for word in text]
16    #text = [Lem.Lemmatize(word) for word in text if not word in stop_words]
17    return text
```

```
In [61]: 1 daniel = ["daniel jackson", 'daniel', 'jackson', 'danny']
2 jack = ['jack', 'o neill', 'neill', 'o.neill', 'o'neill', 'o' neill', 'oneill']
3 samantha = ['samantha carter', 'samantha', 'sam', 'carter']
4 # check for all sam, carter, samanta - they should not exist
5 teal_c = ["teal 'c", "teal'c", 'tealc', 'teal']
6 cameron = ['cameron', 'mitchell', 'michell', 'mitcell']
7 hammond = ['hammond', 'gen hammond', 'gen. hammond', 'general hammond', 'hammond']
8 dr_janet = ['janet', 'dr janet', 'dr. fraiser', 'dr. frasier', 'fraiser', 'frasier']
9 woman = ['woman', 'female']
10 martouf = ['martouf', 'marty']
11 connor = ['conner', 'connor']
12 interlude = []
13 interlude.extend(odd_characters)
14 interlude.extend(long_chatacter_names)
15 interlude.extend(characters_occurring_once)
16
17 name_mappings = {
18     'daniel_jackson': daniel,
19     'jack_o_neill': jack,
20     'samantha_carter': samantha,
21     'teal_c': teal_c,
22     'cameron': cameron,
23     'hammond': hammond,
24     'dr_janet': dr_janet,
25     'woman': woman,
26     'martouf': martouf,
27     'connor': connor,
28     'interlude': interlude
29 }#todo: martouf was added Later
30
31 name_mappings_t = {}
32
33 for character_name, equivalent in name_mappings.items():
34     for e_name in equivalent: name_mappings_t[e_name] = character_name
35 #name_mappings_t
```

In [62]:

Cleaining

In [63]:

In [64]:

In [103]:

In [104]:

Out[104]:

In [106]:

In [105]:

Making Documents

Section the scripts by interludes

In [112]:

In [137]:

In [141]:

...

```
In [142]: 1 #remove interludes
2 all_scripts = all_scripts.loc[all_scripts.character!='interlude']
```

Out[142]:

	character	text	season	episode	doc_id
3	woman	[are, not, you, guys, afraid, of, an, officer,...	1	1	3
6	woman	[does, that, thing, always, do, that]	1	1	5
8	woman	[whatever, it, is, under, the, trap, i, just, ...	1	1	6
11	woman	[im, telling, you, that, thing, is, moving]	1	1	8
15	woman	[i, take, it, this, has, never, happened, before]	1	1	11
...
75203	woolsey	[it, almost, sounds, like, you, might, find, i,...	9	9	34288
75204	daniel_jackson	[no, shakes, head, that, does, not, mean, we, ...	9	9	34288
75205	woolsey	[you, were, right, about, the, risks]	9	9	34288
75207	daniel_jackson	[believe, mei, wish, id, been, wrong, door, cl...	9	9	34289
75208	woolsey	[whispers, me, too]	9	9	34289

40920 rows × 5 columns

```
In [144]: 1 all_scripts.reset_index(inplace=True, drop=True)
```

```
In [145]: 1 all_scripts
```

Out[145]:

	character	text	season	episode	doc_id
0	woman	[are, not, you, guys, afraid, of, an, officer,...	1	1	3
1	woman	[does, that, thing, always, do, that]	1	1	5
2	woman	[whatever, it, is, under, the, trap, i, just, ...	1	1	6
3	woman	[im, telling, you, that, thing, is, moving]	1	1	8
4	woman	[i, take, it, this, has, never, happened, before]	1	1	11
...
40915	woolsey	[it, almost, sounds, like, you, might, find, i...	9	9	34288
40916	daniel_jackson	[no, shakes, head, that, does, not, mean, we, ...	9	9	34288
40917	woolsey	[you, were, right, about, the, risks]	9	9	34288
40918	daniel_jackson	[believe, mei, wish, id, been, wrong, door, cl...	9	9	34289
40919	woolsey	[whispers, me, too]	9	9	34289

40920 rows × 5 columns

Store Data

```
In [146]: 1 # seasons = all_scripts4.season.unique()
2 # episodes = all_scripts4.episode.unique()
3
4 # for season in seasons:
5 #     for episode in episodes:
6 #         df = all_scripts4.loc[(all_scripts4.season == season) &
7 #                               (all_scripts4.episode == episode)]
8 #         df = df[['character', 'text']]
9 #         df.to_csv(clean_data_folder_csv+S+str(season)+"-E"+str(episode)+"-clean.csv")
10 #         df.to_pickle(clean_data_folder_pickle+S+str(season)+"-E"+str(episode)+"-clean.pkl")
```

```
In [146]: 1 all_scripts.to_pickle(clean_data_folder+"/all_data.pkl")
2 all_scripts.to_csv(clean_data_folder+"/all_data.csv")
```

```
In [ ]: 1 # outsource character and gender
2 clean_data_folder = "../clean_data/"
3 with open(clean_data_folder+"all_data_with_gender.pkl", "rb") as f:
4     all_scripts = load(f)
5 characters = all_scripts[["character", "gender", "gender2"]].drop_duplicates()
6 characters.to_pickle(clean_data_folder+"characters_with_gender.pkl")
```