Data Cleaning

In [56]:
```
from pickle import loads, dumps, load
import json
import re
import pandas as pd
from string import punctuation
import nltk
from nltk.stem.porter import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer
#nltk.download('stopwords')
from nltk.corpus import stopwords
from pathlib import Path
import shutil
import os
from numpy import logical_and
from autocorrect import Speller
```

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

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

Steps

- removing '
- remove quotes
- To lowercase
- remove HTML tags
- woman = female
- leerzeichen namen
- Delete emoty 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]:
```
# Odd character found in othr steps of the project
with open('characters_occouring_once.pckl', 'rb') as f1, open('long_chatacter_names.pckl', 'rb') as f2, open('odd_characters.pckl', 'rb') as f3:
        characters_occouring_once = load(f1)
        long_chatacter_names = load(f2)
        odd_characters = load(f3)
odd_characters.remove('child 1')
odd_characters.remove('child 2')
odd_characters.remove('child')
```

In [60]:
```
def clean_text(string:str):
    speller = Speller(lang='en')
    text = string.lower()
    text = text.strip()
    text = re.sub('<[a-zA-A\/]*>','',text)
    text = re.sub('•', "'", text)
    text = re.sub("'re", ' are', text)
    text = re.sub("n't", ' not', text)
    text = re.sub("I'm", 'I am', text)
    text = speller(text)
    text = re.sub("''", ' ', text)
    #source: https://stackoverflow.com/questions/265960/best-way-to-strip-punctuation-from-a-string
    text = text.translate(str.maketrans('', '', punctuation))
    text = text.split()
    text = [speller(word) for word in text]
    #text = [lem.Lemmatize(word) for word in text if not word in stop_words]
    return text
```

In [61]:
```
daniel = ["daniel jackson",'daniel', 'jackson','danny']
jack = ['jack','o• neill',"•neill","o•neill", "o'neill", "o' neill", 'oneill']
samantha = ['samanta carter', 'samanta','sam', 'carter']
# check for all sam, carter, samanta - they should not exist
teal_c = ["teal 'c", "teal'c", 'tealc', 'teal']
cameron = ['cameron', 'mitchell','michell', 'mitcell']
hammond = [ 'hammond', 'gen hammond','gen. hammond', 'general hammond', 'hammond']
dr_janet = ['janet', 'dr janet', 'dr. fraiser', 'dr. frasier', 'fraiser', 'frasier']
woman =  ['woman', 'female']
martouf = ['martouf', 'marty']
connor = ['conner', 'connor']
interlude = []
interlude.extend(odd_characters)
interlude.extend(long_chatacter_names)
interlude.extend(characters_occouring_once)

name_mappings = {
    'daniel_jackson':daniel,
    'jack_o_neill': jack,
    'samantha_carter': samantha,
     'teal_c': teal_c,
     'cameron': cameron,
     'hammond':hammond,
     'dr_janet':dr_janet,
     'woman': woman,
     'martouf':martouf,
    'connor': connor,
    'interlude':interlude
    }#todo: martouf was added later

name_mappings_t = {}

for character_name, equivalent in name_mappings.items():
    for e_name in equivalent: name_mappings_t[e_name] = character_name
#name_mappings_t
```

In [62]:
```
def swap_name(name, mapping):
    for key,value in mapping.items():
        if key in name:
            return(value)
    return(name.strip())
```

Cleaining

In [63]:
```
clean_data_folder = "../clean_data_lda/"
clean_data_folder_csv = "../clean_data_lda/csv/"
clean_data_folder_pickle = "../clean_data_lda/pickle/"
```

In [64]:
```
def purge_folder(clean_data_folder,
                 clean_data_folder_csv,
                 clean_data_folder_pickle):
    try:
        shutil.rmtree(clean_data_folder)
    except: pass
    Path(clean_data_folder).mkdir(parents=True, exist_ok=True)
    Path(clean_data_folder_pickle).mkdir(parents=True, exist_ok=True)
    Path(clean_data_folder_csv).mkdir(parents=True, exist_ok=True)
```

In [103]:
```
purge_folder(clean_data_folder,
             clean_data_folder_csv,
             clean_data_folder_pickle)
```

In [104]:
```
raw_data_folder = "../raw_data/"
files = os.listdir(raw_data_folder)
files.pop(0)
#files
```

Out[104]: 'corpus_dict.pkl'

In [106]:
```
files
```

```
'S1-E2.pkl',
'S1-E20.pkl',
'S1-E21.pkl',
'S1-E3.pkl',
'S1-E4.pkl',
'S1-E5.pkl',
'S1-E6.pkl',
'S1-E7.pkl',
'S1-E8.pkl',
'S1-E9.pkl',
'S2-E1.pkl',
'S2-E10.pkl',
'S2-E11.pkl',
'S2-E12.pkl',
'S2-E13.pkl',
'S2-E14.pkl',
'S2-E15.pkl',
'S2-E16.pkl',
'S2-E17.pkl',
```

In [105]:
```
# Clean all data
for filename in files:
    with open(raw_data_folder+filename, 'rb') as handle:
        df = load(handle)
    #df.columns = ['character', 'text']
    # Data Cleaning steps
    df = df.apply(lambda x: [swap_name(x[0].lower(), name_mappings_t),x[1]], axis=1, result_type='expand')
    df.columns = ['character', 'text']
    df.text.loc[df.character == 'interlude'] = ''
    df = df.apply(lambda x: [x[0],clean_text(x[1])], axis=1, result_type='expand')
    df.columns = ['character', 'text']

    filename_csv = filename.replace('.pkl', '-clean.csv')
    filename_pkl = filename.replace('.pkl', '-clean.pkl')
    df.to_csv(clean_data_folder_csv+filename_csv)
    df.to_pickle(clean_data_folder_pickle+filename_pkl)
```

Making Documents

Section the scripts by interludes

In [112]:
```
# load scripts
all_scripts = pd.DataFrame(columns=['character', 'text', 'season', 'episode'])
clean_files = [x.replace(".pkl", "-clean.pkl") for x in files]
for filename in clean_files:
    with open(clean_data_folder+'pickle/'+filename, 'rb') as handle:
        df = load(handle)
        filename = filename.split('-')
        season = filename[0].replace('S', '')
        df['season'] = int(season)
        episode = filename[1].replace('E', '')
        df['episode'] = int(episode)
        all_scripts = all_scripts.append(df)
```

...

In [137]:
```
all_scripts.reset_index(inplace=True, drop=True)
```

In [141]:
```
#assign document iads
all_scripts['doc_id'] = 0
doc_id = 0

for index, row in all_scripts.iterrows():
    all_scripts.iloc[index,4] = doc_id
    if row['character'] == 'interlude': doc_id+=1
all_scripts
```

...

In [142]:
```
#remove interludes
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]:
```
all_scripts.reset_index(inplace=True, drop=True)
```

In [145]:
```
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 [46]:
```
# seasons = all_scripts4.season.unique()
# episodes = all_scripts4.episode.unique()

# for season in seasons:
#     for episode in episodes:
#         df = all_scripts4.loc[(all_scripts4.season == season) &
#                               (all_scripts4.episode == episode)]
#         df = df[['character', 'text']]
#         df.to_csv(clean_data_folder_csv+"S"+str(season)+"-E"+str(episode)+"-clean.csv")
#         df.to_pickle(clean_data_folder_pickle+"S"+str(season)+"-E"+str(episode)+"-clean.pkl")
```

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

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