Differences in Male and Female Speech

Männer vs. Frauen -> über gesamte Serie, Episoden, Staffeln

Männer haben einen höheren Redeanteil als Frauen.

Männer und Frauen reden über unterschiedliche Themen.

Bechtel Test.

In [2]:
```python
import pandas as pd
import matplotlib.pyplot as plt
from pickle import load, dump
import os
import gender_guesser.detector as gender
import seaborn as sns
import numpy as np

```

Load the Data

In [3]:
```python
clean_data_folder = "../clean_data_"
filenames = os.listdir(clean_data_folder)
os.listdir(clean_data_folder)
```

Out[3]:
```
['all_data.csv',
 'all_data.pkl',
 'all_data_with_gender.csv',
 'all_data_with_gender.pkl',
 'csv',
 'pickle',
 'unknown_names.csv']
```

In [4]:
```python
with open(clean_data_folder+"/all_data.pkl", "rb") as f:
    all_scripts = load(f)
```

In [5]:
```python
all_scripts
```

Out[5]:

	character	text	season	episode	word_count
0	airman	[oh, man, this, hands, as, lousy, as, this, de...	1	1	15
2	airman	[seven, to, the, deuce, nothing, there, boss, ...	1	1	32
3	woman	[are, not, you, guys, afraid, of, an, officer,...	1	1	13
4	officer	[trust, me, nobody, ever, comes, down, here, b...	1	1	9
6	woman	[does, that, thing, always, do, that]	1	1	6
...	...	...	...	...	...
396	woolsey	[it, almost, sounds, like, you, might, find, i...	9	9	13
397	daniel_jackson	[no, shakes, head, that, does, not, mean, we, ...	9	9	23
398	woolsey	[you, were, right, about, the, risks]	9	9	6
400	daniel_jackson	[believe, mei, wish, id, been, wrong, door, cl...	9	9	8
401	woolsey	[whispers, me, too]	9	9	3

59843 rows × 5 columns

Determine Gender

Approach 1: Gender Guesser Library

In [6]:
```python
d = gender.Detector()
#print(d.get_gender(u"bob"))
```

In [7]:
```python
# Add an extra column to only include the capitalized first name
# this is the only format the gender guesser accepts
all_scripts['name_split'] = all_scripts.character.apply(lambda x: x.split("_")[0].capitalize())
all_scripts['gender'] = all_scripts.name_split.apply(lambda x: d.get_gender(x))
```

In [8]:
```python
all_scripts
```

Out[8]:

	character	text	season	episode	word_count	name_split	gender
0	airman	[oh, man, this, hands, as, lousy, as, this, de...	1	1	15	Airman	unknown
2	airman	[seven, to, the, deuce, nothing, there, boss, ...	1	1	32	Airman	unknown
3	woman	[are, not, you, guys, afraid, of, an, officer,...	1	1	13	Woman	unknown
4	officer	[trust, me, nobody, ever, comes, down, here, b...	1	1	9	Officer	unknown
6	woman	[does, that, thing, always, do, that]	1	1	6	Woman	unknown
...	...	...	...	...	...	...	...
396	woolsey	[it, almost, sounds, like, you, might, find, i...	9	9	13	Woolsey	unknown
397	daniel_jackson	[no, shakes, head, that, does, not, mean, we, ...	9	9	23	Daniel	male
398	woolsey	[you, were, right, about, the, risks]	9	9	6	Woolsey	unknown
400	daniel_jackson	[believe, mei, wish, id, been, wrong, door, cl...	9	9	8	Daniel	male
401	woolsey	[whispers, me, too]	9	9	3	Woolsey	unknown

59843 rows × 7 columns

In [9]:
```python
all_scripts.gender.unique()
```

Out[9]:
```
array(['unknown', 'female', 'male', 'andy', 'mostly_female',
       'mostly_male'], dtype=object)
```

In [10]:
```python
# Count genders
all_scripts[['character', 'gender']].drop_duplicates().gender.value_counts()
```

Out[10]:
```
unknown          644
male              99
female            43
mostly_male       12
andy               7
mostly_female      6
Name: gender, dtype: int64
```

811 characters in total, 644 (79,4%) genders could not be guessed, 142 (17,5%) were guesses male or female, 18 (2,2%) were mostly male or female, and 7 (0,8%) names work for both genders.

For this relatively poor performance we chose to label the genders by hand. We will hower use the results to test alternatice approches that are more sustainable.

In [11]:
```python
# Inspect the names who's genders could not be guessed
unknown_names = list(all_scripts.loc[all_scripts.gender=="unknown"].character.unique())
unknown_names.sort()
unknown_names
```

Out[11]:
```
['"auto destruct in',
 "'gateroom",
 '.',
 'abg',
 'adal',
 'administrator',
 'aegir',
 'ag',
 'ahc',
 'airman',
 'airmen',
 'albant',
 'albran',
 'alebran',
 'alekos',
 'alien planet',
 'all',
 'alpha site',
 'alpha site infirmary',
```

In [12]:
```python
# Inspect the names of characters who's names were guessed
charachters_with_genders = all_scripts.loc[all_scripts.gender!="unknown"][['character', 'gender']].drop_duplicates()
print(charachters_with_genders.sort_values('character').to_string())
```

```
          character         gender
7               abu           male
365            aide         female
170          aiyana         female
54               al           male
41             alar           male
191          aldwin           male
293           alien         female
230            ally         female
13             andy           male
128            anna         female
29             aris           male
124            aron           male
351         barrett           male
21             bert           male
61             bill           male
380             boy           male
0              boyd           male
22           brenna         female
```

Check accuracy of names guessed male or female

We are sampleling 20 names to test the accuracy of the gender-guesser

In [13]:
```python
sample = charachters_with_genders.loc[(charachters_with_genders.gender == "female") |\
                                      (charachters_with_genders.gender == "male")].sample(20)
```

In [14]:
```python
sample
```

Out[14]:

	character	gender
51	harper	male
106	lawrence	male
116	rogelio	male
47	krista	female
199	shauna	female
196	oma	female
18	glen	male
18	maynard	male
35	kieran	male
253	murphy	male
142	raphael	male
128	anna	female
75	thor	male
55	harlow	male
112	delores	female
95	kendra	female
44	simon	male
1	leda	female
251	seth	male
81	marine	female

Mostly male or female:

3 out of 6 mostly female genderes are in fact male:

kennedy, MacKenzie, Solen

and 1 of the 12 mostly male characters is in fact female

Harley → not even a character? acrually Sara O'Neil???

Male or female

3 were guessed wrong:

- Lindsey Novak → female
- Hale --> Male
- kegan--> female

9 character's gender could not be verified because: not found, the gender not mentioned, or characters with both genders exist.

- delores, glen, --> character cannot be found
- nesa --> gender not known / no further information
- tobias, jones, Cole, johnson --> characters with geneders exist
- raphael, calvin, --> characters could not be found, but usually male

Conclusion

For the clear male or female cases the gender guesser has an accuracy of 55% for the names that are mostly male or female, the accuracy is about 78% but there is a clear prevalence of male characters and female names are predicted wrong more often than male names.

For this reason we also detremined these genders by hand. However we are investigating better methods, because this does not scale well.

Approaches: Gender guesser, NLTK, Hand, Code. Which one yields the best results, is the most feasable,

Further Data Cleaning Needed on planet, room, sgcm the, to be continued fix more typos → similar names???

In [15]:
```
mostly_genders_p = 18/167

mostly_genders_p*(12/18)+(1-mostly_genders_p)*0.4
```
Out[15]:
```
0.42874251497005994
```

Approach 2: Assign genders by hand

In [16]:
```
# We exported all the unknown names and added the gender by hand
# After we realized the gender guesser is not accurate we redid the the gender assignment of those characters by hand
```

In [17]:
```
#charachters_with_genders.to_csv('unknown_names2.csv')
```

In [18]:
```
unknown_names_genderized = pd.read_excel('unknown_names_gendered.xlsx')
unknown_names_genderized = unknown_names_genderized[[0, 'Unnamed: 2']]
unknown_names_genderized.columns = ['character', 'gender']
# Some conversion error fix
unknown_names_genderized = unknown_names_genderized.replace(to_replace=r'Â’', value='\x92', regex=True)
print(f"number of records: {len(unknown_names_genderized)}")
unknown_names_genderized.head()
```
```
number of records: 644
```
Out[18]:

	character	gender
0	airman	male
1	woman	female
2	officer	male
3	apophis	male
4	soldier	male

In [19]:
```
# Remaining names genderized that were previously assigned by the gender API
names_genderized = pd.read_csv('unknown_names_gendered2.csv')
print(f"number of records: {len(names_genderized)}")
names_genderized.head()
```
```
number of records: 167
```
Out[19]:

	character	gender
0	samantha_carter	female
1	jack_o_neill	male
2	warner	male
3	daniel_jackson	male
4	boy	male

In [20]:
```
# Combine
all_names_genderized= names_genderized.append(unknown_names_genderized)
all_names_genderized=all_names_genderized.reset_index(drop = True)
print(f"number of records: {len(all_names_genderized)}")
all_names_genderized.head()
```
```
number of records: 811

C:\Users\debor\AppData\Local\Temp\ipykernel_14852\2917432786.py:2: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.conca
t instead.
  all_names_genderized= names_genderized.append(unknown_names_genderized)
```
Out[20]:

	character	gender
0	samantha_carter	female
1	jack_o_neill	male
2	warner	male
3	daniel_jackson	male
4	boy	male

In [21]:
```
all_names_genderized.gender.value_counts()
```
Out[21]:
```
male        449
unknown     193
female      136
both         10
neutral      10
maöe          3
unknoown      2
unkown        2
unkniwn       1
unknpen       1
unknwon       1
weiblich      1
uknown        1
unknowm       1
Name: gender, dtype: int64
```

In [22]:
```
fix_typos = {}
fix_typos['andy'] = 'neutral'
fix_typos['unknoown']  = fix_typos['unkniwn'] =\
fix_typos['unknpen']  = fix_typos['unkown']  = \
fix_typos['uknown']  = fix_typos['unknowm'] =\
fix_typos['unknwon'] =  'unknown'

fix_typos['maöe'] = 'male'
fix_typos['weiblich'] = 'female'
fix_typos
```
Out[22]:
```
{'andy': 'neutral',
 'unknoown': 'unknown',
 'unkniwn': 'unknown',
 'unknpen': 'unknown',
 'unkown': 'unknown',
 'uknown': 'unknown',
 'unknowm': 'unknown',
 'unknwon': 'unknown',
 'maöe': 'male',
 'weiblich': 'female'}
```

In [23]:
```python
for key in fix_typos.keys():
    all_names_genderized.loc[all_names_genderized['gender'] == key, 'gender'] = fix_typos[key]
all_names_genderized.gender.value_counts()
```

Out[23]:
```
male       452
unknown    202
female     137
both        10
neutral     10
Name: gender, dtype: int64
```

In [24]:
```python
# Add column with gender to df
gender_dict = {}
for row in all_names_genderized.iterrows():
    gender_dict[row[1]['character']] = row[1]['gender']
all_scripts['gender'] = all_scripts['character'].apply(lambda x: gender_dict[x])
all_scripts.head()
```

Out[24]:

	character	text	season	episode	word_count	name_split	gender
0	airman	[oh, man, this, hands, as, lousy, as, this, de...	1	1	15	Airman	male
2	airman	[seven, to, the, deuce, nothing, there, boss, ...	1	1	32	Airman	male
3	woman	[are, not, you, guys, afraid, of, an, officer,...	1	1	13	Woman	female
4	officer	[trust, me, nobody, ever, comes, down, here, b...	1	1	9	Officer	male
6	woman	[does, that, thing, always, do, that]	1	1	6	Woman	female

In [25]:
```python
all_scripts.gender.unique()
```

Out[25]:
```
array(['male', 'female', 'unknown', 'both', 'neutral'], dtype=object)
```

In [26]:
```python
# Add another column with the gender "unknowmn", "both" and "neutral" summarized as "unclear"
gender2_dict = {'male': 'male',
                'female':'female',
                'unknown':'unclear',
                'both':'unclear',
                'neutral':'unclear'}

all_scripts['gender2'] = all_scripts['gender'].apply(lambda x: gender2_dict[x])
all_scripts.loc[all_scripts.gender2=='unclear'].sample(20)
```

Out[26]:

	character	text	season	episode	word_count	name_split	gender	gender2
101	danny	[jack]	5	7	1	Danny	unknown	unclear
301	sgc briefing room	[hammond, and, sg1, are, around, the, table, a...	6	7	8	Sgc briefing room	unknown	unclear
149	rc	[all, of, your, teams, have, arrived, safely, ...	5	14	16	Rc	unknown	unclear
146	davis	[sir, we, lost, the, signal]	5	4	5	Davis	both	unclear
297	dixon	[understood, sir]	7	17	2	Dixon	both	unclear
102	davis	[this, is, stargate, command, calling, doctor,...	8	5	11	Davis	both	unclear
87	gh	[apophis]	5	1	1	Gh	unknown	unclear
148	principal	[air, force, people, he, steps, aside, to, sho...	8	19	25	Principal	unknown	unclear
6	cole	[my, god, they, are, egyptian, that, does, not...	1	13	17	Cole	both	unclear
35	danny	[we, surrender]	5	1	2	Danny	unknown	unclear
292	landry	[walks, out, you, got, something, else, for, m...	9	7	9	Landry	both	unclear
79	davis	[the, goauld, are, a, predatory, species, they...	6	17	61	Davis	both	unclear
54	ven	[a, symbol, of, our, new, unity]	7	14	6	Ven	unknown	unclear
304	fisher	[angry, excuse, me, the, woman, beside, the, s...	9	4	13	Fisher	unknown	unclear
331	jaffa	[my, lord, a, cargo, ship, was, detected, exit...	5	16	18	Jaffa	unknown	unclear
225	ag	[its, drawing, energy, from, the, ionization, ...	4	6	15	Ag	unknown	unclear
181	scientist 1	[we, have, modified, the, android, body, recov...	9	1	32	Scientist 1	unknown	unclear
173	davis	[no, pupolsion, of, any, kind, and, this, atte...	4	12	31	Davis	both	unclear
311	danny	[well, we, have, to, give, them, the, option, ...	5	5	12	Danny	unknown	unclear
19	davis	[the, russian, team, major, and, they, are, re...	6	16	11	Davis	both	unclear

In [27]:
```python
# Save

all_scripts.to_pickle(clean_data_folder+"/all_data_with_gender.pkl")
all_scripts.to_csv(clean_data_folder+"/all_data_with_gender.csv")
```

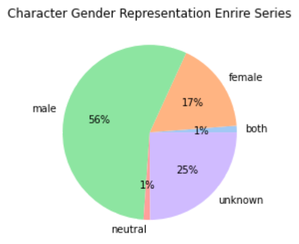
Plot results

In [28]:
```python
word_count_gender = all_scripts[['gender','word_count']].groupby(['gender']).sum()
word_count_gender2 = all_scripts[['gender2','word_count']].groupby(['gender2']).sum()
```

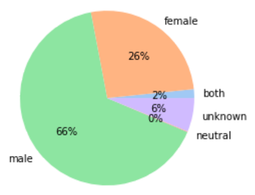
In [29]:
```python
character_count_gender = all_scripts[['character', 'gender']].drop_duplicates().groupby('gender').count()
character_count_gender2 = all_scripts[['character', 'gender2']].drop_duplicates().groupby('gender2').count()
```

All

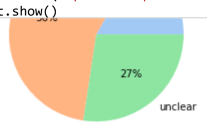
In [30]:
```python
#source: https://www.statology.org/seaborn-pie-chart/
data = character_count_gender.character
labels = character_count_gender.index
colors = sns.color_palette('pastel')[0:5]
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
plt.title("Character Gender Representation Enrire Series")
plt.show()
```



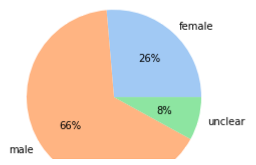
In [31]:
```python
data = word_count_gender.word_count
labels = word_count_gender.index
colors = sns.color_palette('pastel')[0:5]
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
plt.show()
```



In [32]:
```python
data = character_count_gender2.character
labels = character_count_gender2.index
colors = sns.color_palette('pastel')[0:5]
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
plt.title("Character Gender Representation Entire Series")
plt.show()

data = word_count_gender2.word_count
labels = word_count_gender2.index
colors = sns.color_palette('pastel')[0:5]
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
plt.title("Speech Proportion of Genders Entire Series")
plt.show()
```



Speech Proportion of Genders Entire Series



Per Season

In [33]:
```python
season_word_count2 = all_scripts[['season', 'word_count']].groupby(['season']).sum()
season_word_count2.head()
```

Out[33]:
```
        word_count
season
     1       76413
     2       77922
     3       72920
     4       82816
     5       89922
```

In [34]:
```python
word_count_gender = all_scripts[['gender2', 'season','word_count']]
data = word_count_gender.groupby(['gender2', 'season']).sum()
data.head()
```

Out[34]:
```
                word_count
gender2 season
 female      1       20546
             2       23468
             3       18413
             4       23033
             5       23383
```

In [35]:
```python
data = data.join(season_word_count2, on='season', lsuffix='', rsuffix='_season')
data.head()
```

Out[35]:
```
                word_count  word_count_season
gender2 season
 female      1       20546              76413
             2       23468              77922
             3       18413              72920
             4       23033              82816
             5       23383              89922
```

In [36]:
```
data['percentage'] = data['word_count']/data['word_count_season']
data.head()
```
Out[36]:

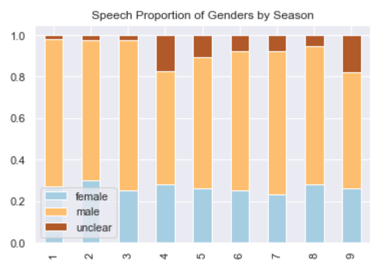
		word_count	word_count_season	percentage
gender2	**season**			
female	**1**	20546	76413	0.268881
	2	23468	77922	0.301173
	3	18413	72920	0.252510
	4	23033	82816	0.278123
	5	23383	89922	0.260036

In [37]:
```
data2=data.reset_index()
```
In [38]:
```
data3 = data2[['percentage','season', 'gender2']]
```
In [39]:
```
# Transform data for staked bar plot

season_plot_df=pd.DataFrame(index=['female', 'male', 'unclear'])

for season in range(1,10,1):
    season_data = data3.loc[data3.season==season]
    season_data = season_data[['gender2', 'percentage']]
    season_data.columns = ['gender2', str(season)]
    season_data = season_data.set_index('gender2')
    season_plot_df = season_plot_df.join(season_data)

season_plot_df
```
Out[39]:

	1	2	3	4	5	6	7	8	9
female	0.268881	0.301173	0.252510	0.278123	0.260036	0.249515	0.231130	0.280635	0.262495
male	0.713805	0.675291	0.720996	0.549121	0.632026	0.674437	0.692569	0.664830	0.556386
unclear	0.017314	0.023536	0.026495	0.172756	0.107938	0.076048	0.076301	0.054535	0.181119

In [40]:
```
season_plot_df.T
```
Out[40]:

	female	male	unclear
1	0.268881	0.713805	0.017314
2	0.301173	0.675291	0.023536
3	0.252510	0.720996	0.026495
4	0.278123	0.549121	0.172756
5	0.260036	0.632026	0.107938
6	0.249515	0.674437	0.076048
7	0.231130	0.692569	0.076301
8	0.280635	0.664830	0.054535
9	0.262495	0.556386	0.181119

In [41]:
```
sns.set()
plot = season_plot_df.T.plot(kind='bar',
                         stacked=True,
                         colormap= plt.cm.get_cmap('Paired'),
                         title="Speech Proportion of Genders by Season")
```


Per Episode

In [42]:
```
all_scripts#[['gender2','word_count']]
```
Out[42]:

	character	text	season	episode	word_count	name_split	gender	gender2
0	airman	[oh, man, this, hands, as, lousy, as, this, de...	1	1	15	Airman	male	male
2	airman	[seven, to, the, deuce, nothing, there, boss, ...	1	1	32	Airman	male	male
3	woman	[are, not, you, guys, afraid, of, an, officer,...	1	1	13	Woman	female	female
4	officer	[trust, me, nobody, ever, comes, down, here, b...	1	1	9	Officer	male	male
6	woman	[does, that, thing, always, do, that]	1	1	6	Woman	female	female
...	...	...	...	...	...	...	...	...
396	woolsey	[it, almost, sounds, like, you, might, find, i...	9	9	13	Woolsey	male	male
397	daniel_jackson	[no, shakes, head, that, does, not, mean, we, ...	9	9	23	Daniel	male	male
398	woolsey	[you, were, right, about, the, risks]	9	9	6	Woolsey	male	male
400	daniel_jackson	[believe, mei, wish, id, been, wrong, door, cl...	9	9	8	Daniel	male	male
401	woolsey	[whispers, me, too]	9	9	3	Woolsey	male	male

59843 rows × 8 columns

In [43]:
```
def season_episode_plot(season):
    season_data = all_scripts1.loc[all_scripts1['season'] == season]
    episode_word_count = season_data[['episode', 'word_count']].groupby(['episode']).sum()
    season_data_and_episode_wc = season_data.join(episode_word_count, on='episode',  lsuffix='_character', rsuffix='_episode')
    season_data_and_episode_wc['percentage'] = season_data_and_episode_wc['word_count_character']/season_data_and_episode_wc['word_count_episode']
    data = season_data_and_episode_wc[['gender','episode', 'percentage']].groupby(['episode','gender']).sum()
    sns.lineplot(data=data, x="episode", y="percentage", hue='gender')
```

In [44]:
```
episode_word_count = all_scripts[['season','episode', 'word_count']].groupby(['episode','season']).sum()#.reset_index()
episode_word_count_gender = all_scripts[['season','episode', 'word_count', 'gender2']].groupby(['episode','season', 'gender2']).sum().reset_index('gender2')
episode_wc = episode_word_count_gender.merge(episode_word_count, on=['episode','season'],  how='inner', suffixes=('_gender', '_episode'))
episode_wc['wc_percent'] = episode_wc['word_count_gender']/episode_wc['word_count_episode']

episode_wc
```

Out[44]:

		gender2	word_count_gender	word_count_episode	wc_percent
episode	**season**				
1	**1**	female	1194	6472	0.184487
	1	male	5218	6472	0.806242
	1	unclear	60	6472	0.009271
	2	female	794	3174	0.250158
	2	male	2309	3174	0.727473
...	**...**	...	...	...	...
22	**6**	male	2773	3166	0.875869
	6	unclear	114	3166	0.036008
	7	female	1144	4813	0.237690
	7	male	3368	4813	0.699771
	7	unclear	301	4813	0.062539

521 rows × 4 columns

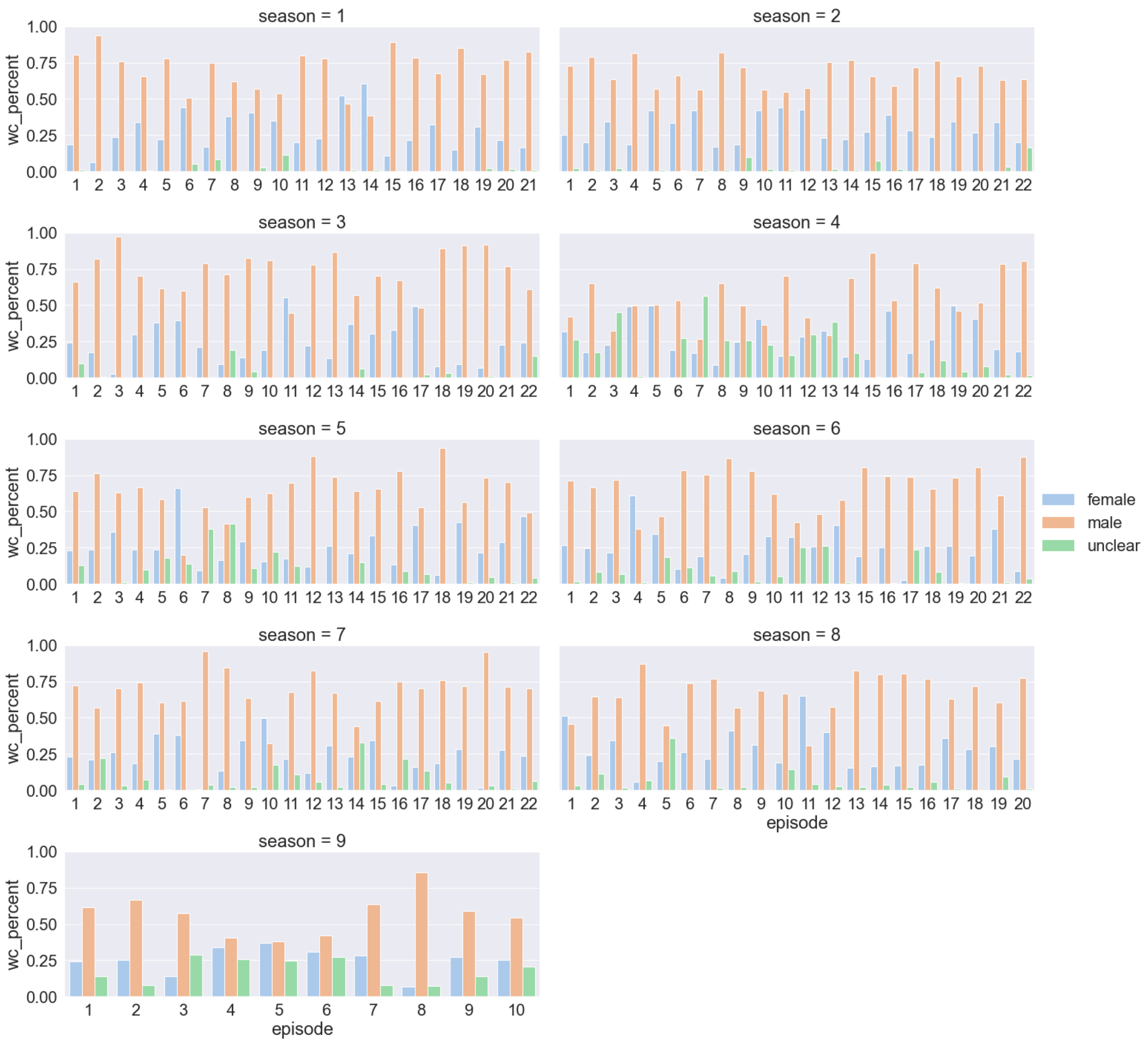
In [45]:
```
sns.color_palette('pastel')[0:1]
```

Out[45]:
```
[(0.6313725490196078, 0.788235294117647, 0.9568627450980393)]
```

In [46]:
```
sns.set(font_scale=2)

g = sns.FacetGrid(episode_wc.reset_index(),
                  col="season",
                  col_wrap=2,
                  height=4,
                  ylim=(0, 1),
                 sharex=False, aspect=2.5)
g.map(sns.barplot,
      "episode",
      "wc_percent",
      'gender2',
      palette=sns.color_palette('pastel')[0:3],
      ci=None)
g.add_legend()

```

```
C:\Users\debor\anaconda3\lib\site-packages\seaborn\axisgrid.py:645: UserWarning: Using the barplot function without specifying `order` is likely to produce an incorrect plot.
  warnings.warn(warning)
C:\Users\debor\anaconda3\lib\site-packages\seaborn\axisgrid.py:650: UserWarning: Using the barplot function without specifying `hue_order` is likely to produce an incorrect plot.
  warnings.warn(warning)
```

Out[46]:
```
<seaborn.axisgrid.FacetGrid at 0x1a282f91730>
```



Investigate Speech amount (Actually additional EDA)

In [47]:
```
word_counts = all_scripts[['season', 'episode', 'word_count']]
word_counts['episode_str']=word_counts['episode'].apply(lambda x: '0'+str(x) if len(str(x))==1 else str(x))
word_counts['season_episode_no'] = word_counts['season'].apply(lambda x: str(x))+word_counts['episode_str']
word_counts['season_episode_no'] = word_counts['season_episode_no'].apply(lambda x: int(x))
```

```
C:\Users\debor\AppData\Local\Temp\ipykernel_14852\2912189056.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  word_counts['episode_str']=word_counts['episode'].apply(lambda x: '0'+str(x) if len(str(x))==1 else str(x))
C:\Users\debor\AppData\Local\Temp\ipykernel_14852\2912189056.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  word_counts['season_episode_no'] = word_counts['season'].apply(lambda x: str(x))+word_counts['episode_str']
C:\Users\debor\AppData\Local\Temp\ipykernel_14852\2912189056.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  word_counts['season_episode_no'] = word_counts['season_episode_no'].apply(lambda x: int(x))
```

In [48]:
```
data = word_counts[['season','word_count', 'episode']].groupby(['season','episode']).sum().reset_index()
data
```

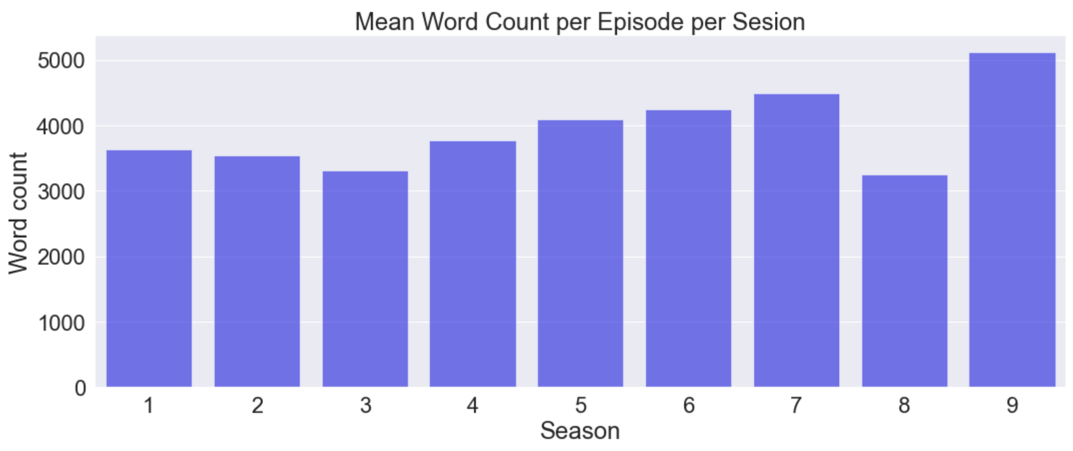
Out[48]:

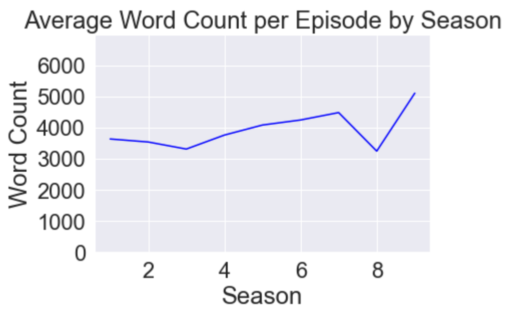
	season	episode	word_count
0	1	1	6472
1	1	2	4316
2	1	3	3105
3	1	4	3687
4	1	5	3324
...	...	...	...
177	9	6	5469
178	9	7	5130
179	9	8	3964
180	9	9	4859
181	9	10	5237

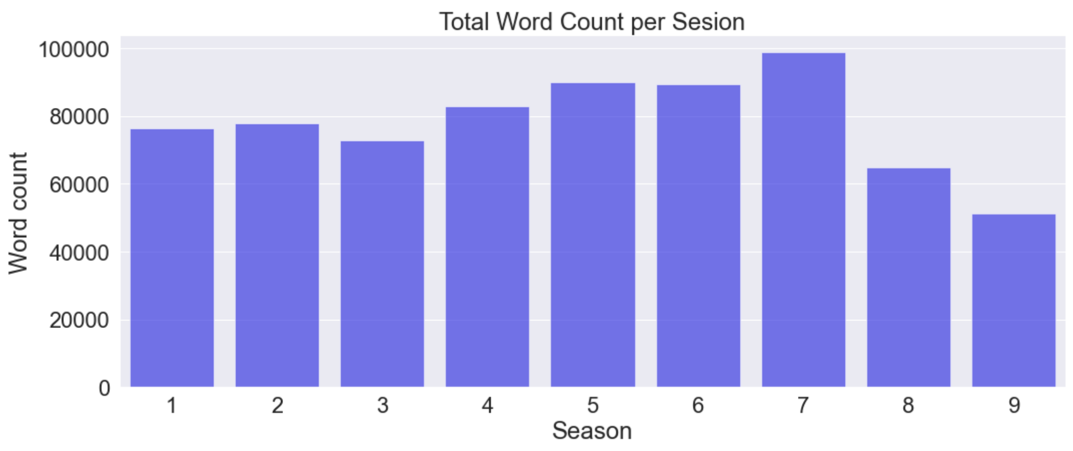
182 rows × 3 columns

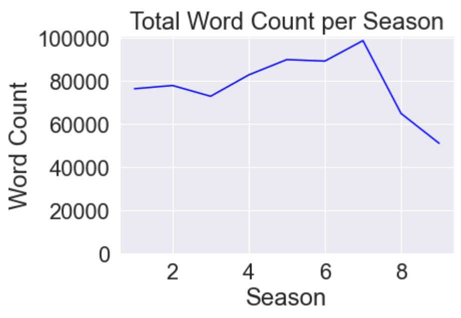
In [49]:
```
#word_counts.season.apply(lambda x: int(x))
# Draw a nested barplot by species and sex
g = sns.catplot(
    data=data[['season','word_count']], kind="bar",
    x="season", y="word_count",
    ci=None, alpha=.6, height=6, aspect=2.5, estimator=np.mean,color='blue'
)
g.set(title='Mean Word Count per Episode per Sesion')
g.set_axis_labels("Season", "Word count")

# ax.fig.suptitle('Title')
# g.despine(left=True)
# #g.set_axis_labels("", "Body mass (g)")

```

Out[49]:
```
<seaborn.axisgrid.FacetGrid at 0x1a2831b5190>
```

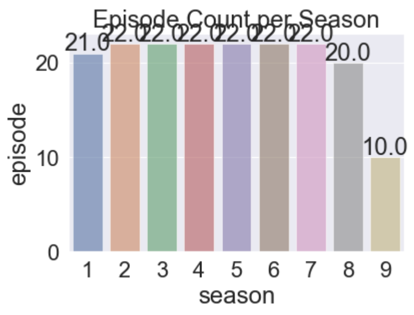


In [50]:
```
g = sns.lineplot(
    data=data[['season','word_count']],
    x="season", y="word_count",
    color='blue',
    ci=None,
    estimator=np.mean
)
g.set(title='Average Word Count per Episode by Season',
     xlabel='Season',
      ylabel='Word Count',
     yticks=[x for x in range(0,7000,1000)],
     ylim=[0,7000])

```
Out[50]:
```
[Text(0.5, 1.0, 'Average Word Count per Episode by Season'),
 Text(0.5, 0, 'Season'),
 Text(0, 0.5, 'Word Count'),
 [<matplotlib.axis.YTick at 0x1a283c6a190>,
  <matplotlib.axis.YTick at 0x1a28437c9d0>,
  <matplotlib.axis.YTick at 0x1a283b51cd0>,
  <matplotlib.axis.YTick at 0x1a283b58280>,
  <matplotlib.axis.YTick at 0x1a2843778e0>,
  <matplotlib.axis.YTick at 0x1a283b51730>,
  <matplotlib.axis.YTick at 0x1a283c81430>],
 (0.0, 7000.0)]
```


In [51]:
```
g = sns.catplot(
    data=data[['season','word_count']], kind="bar",
    x="season", y="word_count",
    ci=None, alpha=.6, height=6, aspect=2.5, estimator=np.sum,color='blue'
)
g.set(title='Total Word Count per Sesion')
g.set_axis_labels("Season", "Word count")
```
Out[51]:
```
<seaborn.axisgrid.FacetGrid at 0x1a282f91370>
```


In [52]:
```
g = sns.lineplot(
    data=data[['season','word_count']],
    x="season", y="word_count",
    color='blue',
    ci=None,
    estimator=np.sum
)
g.set(title='Total Word Count per Season',
     xlabel='Season',
      ylabel='Word Count',
    yticks=[x for x in range(0,110000,20000)])

```
Out[52]:
```
[Text(0.5, 1.0, 'Total Word Count per Season'),
 Text(0.5, 0, 'Season'),
 Text(0, 0.5, 'Word Count'),
 [<matplotlib.axis.YTick at 0x1a283be1eb0>,
  <matplotlib.axis.YTick at 0x1a283be1730>,
  <matplotlib.axis.YTick at 0x1a283c0ab80>,
  <matplotlib.axis.YTick at 0x1a283c1c880>,
  <matplotlib.axis.YTick at 0x1a283c200a0>,
  <matplotlib.axis.YTick at 0x1a283bcd6a0>]]
```


In [53]:
```python
g = sns.barplot(
    data=data[['season','episode']],
    x="season", y="episode",
    ci=None, alpha=.6, estimator=np.max,
)
g.set(title='Episode Count per Season')
# g.set_axis_labels("Season", "Episode count")


for p in g.patches:
    g.annotate(format(p.get_height(), '.1f'),
                   (p.get_x() + p.get_width() / 2., p.get_height()),
                   ha = 'center', va = 'center',
                   xytext = (0, 9),
                   textcoords = 'offset points')
```



In [ ]:
```
```

In [ ]:
```
```