Full Corpus Sentiment Analysis

DS 5001: Exploratory Text Analytics

Cecily Wolfe (cew4pf)

Spring 2022

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
from IPython.display import display, HTML

import nltk
from nltk.stem.porter import PorterStemmer
from nltk.stem.snowball import SnowballStemmer
from nltk.stem.lancaster import LancasterStemmer

from bow_tfidf_pca import create_bow, get_tfidf, get_pca

import itertools
```

In [2]:
```
sns.set()
```

In [3]:
```
OHCO = ['book_id', 'chap_id', 'para_num', 'sent_num', 'token_num']
emo_cols = "anger anticipation disgust fear joy sadness surprise trust polarity"
```

In [4]:
```
SENTS = OHCO[:4]
PARAS = OHCO[:3]
CHAPS = OHCO[:2]
BOOKS = OHCO[:1]
```

Get Data

In [5]:
```
# read in csv files

dickens_LIB = pd.read_csv('dickens_pre_LIB.csv')

dickens_CORPUS = pd.read_csv('dickens_pre_CORPUS.csv')

twain_LIB = pd.read_csv('twain_pre_LIB.csv')

twain_CORPUS = pd.read_csv('twain_pre_CORPUS.csv')
```

LIB Table

In [6]:
```
# combined LIB
LIB = pd.concat([dickens_LIB, twain_LIB]).set_index(BOOKS).sort_index()
```

In [7]:
```
LIB['label'] = LIB.apply(lambda x: "{}_{}_{}".format(x.author, x.title.replace('
```

In [8]:
```
LIB
```

Out[8]:

book_id	source_file_path	title	chap_regex	author	
70	Twain/70-what_is_man.txt	what is man	WHAT IS MAN?|THE DEATH OF JEAN|THE TURNING-POI...	twain	fi
74	Twain/74-the_adventures_of_tom_sawyer.txt	the adventures of tom sawyer	^\s*CHAPTER\s*[IVXLCM]+$	twain	l
76	Twain/76-the_adventures_of_huckleberry_finn.txt	the adventures of huckleberry finn	^\s*CHAPTER\s*(?:[IVXLCM]+\.|THE LAST)$	twain	l
86	Twain/86-a_connecticut_yankee_in_king_arthurs_...	a connecticut yankee in king arthurs court	^\s*(?:PREFACE|A WORD OF EXPLANATION|THE STRAN...	twain	l
91	Twain/91-tom_sawyer_abroad.txt	tom sawyer abroad	CHAPTER\s[IVXLCM]+\.	twain	l
...	...	...	...	...	
35536	Dickens/35536-the_poems_and_verses_of_charles_...	the poems and verses of charles dickens	THE VILLAGE COQUETTES |*THELAMPLIGHTER* |SONGS ...	dickens	st
60900	Twain/60900-merry_tales.txt	merry tales	^THE PRIVATE HISTORY OF A CAMPAIGN THAT FAILED...	twain	st
61522	Twain/61522-the_1000000_bank_note.txt	the 1000000 bank note	^_THE £1,000,000 BANK-NOTE_$|^_METNAL TELEGRAP...	twain	st
62636	Twain/62636-to_the_person_sitting_in_darkness.txt	to the person sitting in darkness	^Extending the Blessings	twain	fi
62739	Twain/62739-king_leopolds_soliloquy.txt	king leopolds soliloquy	^(\[_Throws down pamphlets which he has|Footnote)	twain	st

95 rows × 10 columns

CORPUS Table

In [9]:
```
# combined corpus

CORPUS = pd.concat([dickens_CORPUS, twain_CORPUS]).set_index(OHCO)

# remove NaN values
CORPUS = CORPUS[~CORPUS.term_str.isna()]
```

In [10]:
```
CORPUS
```

Out[10]:

					pos_tuple	pos	token_str	term_str
book_id	chap_id	para_num	sent_num	token_num				
98	1	0	0	0	('The', 'DT')	DT	The	the
				1	('Period', 'NN')	NN	Period	period
		1	0	0	('It', 'PRP')	PRP	It	it
				1	('was', 'VBD')	VBD	was	was
				2	('the', 'DT')	DT	the	the
...	...	...	...	...	...	...	...	...
62739	6	13	0	8	("Leopold's", 'NNP')	NNP	Leopold's	leopolds
				9	('Soliloquy,', 'NNP')	NNP	Soliloquy,	soliloquy
				10	('by', 'IN')	IN	by	by
				11	('Mark', 'NNP')	NNP	Mark	mark
				12	('Twain', 'NNP')	NNP	Twain	twain

7940320 rows × 4 columns

VOCAB Table

In [11]:
```
VOCAB = pd.read_csv('full_VOCAB.csv').set_index('term_str')
```

In [12]:
```
VOCAB.head()
```

Out[12]:

	term_rank	n	n_chars	p	i	max_pos	n_pos	cat_pos	stop	ste
term_str										

term_str	term_rank	n	n_chars	p	i	max_pos	n_pos	cat_pos	stop	ste
the	1	418963	3	0.052764	4.244302	DT	22	{'PRP', 'FW', 'RB', 'NN', 'JJS', 'NNP', 'VBZ',...	1	
and	2	310105	3	0.039054	4.678368	CC	20	{'PRP', 'FW', 'RB', 'PDT', 'NN', 'NNP', 'VBZ',...	1	
of	3	218996	2	0.027580	5.180221	IN	19	{'PRP', 'FW', 'RB', 'PDT', 'NN', 'NNP', 'VBZ',...	1	
to	4	206700	2	0.026032	5.263587	TO	23	{'WDT', 'FW', 'RB', 'PDT', 'NN', 'NNP', 'VBZ',...	1	
a	5	189310	1	0.023842	5.390375	DT	21	{'RBR', 'PRP', 'FW', 'RB', 'NN', 'NNP', 'VBZ',...	1	

In [13]:
```
# CHAPTERS = CORPUS.groupby(OHCO[:2]+['term_str']).term_str.count().unstack()
# VOCAB['df'] = CHAPTERS.count()
# VOCAB['dfidf'] = VOCAB.df * np.log2(len(CHAPTERS)/VOCAB.df)
```

In [14]:
```
VOCAB.head()
```

Out[14]:

term_str	term_rank	n	n_chars	p	i	max_pos	n_pos	cat_pos	stop	ste
the	1	418963	3	0.052764	4.244302	DT	22	{'PRP', 'FW', 'RB', 'NN', 'JJS', 'NNP', 'VBZ',...	1	

term_str	term_rank	n	n_chars	p	i	max_pos	n_pos	cat_pos	stop	ste
and	2	310105	3	0.039054	4.678368	CC	20	{'PRP', 'FW', 'RB', 'PDT', 'NN', 'NNP', 'VBZ',...	1	
of	3	218996	2	0.027580	5.180221	IN	19	{'PRP', 'FW', 'RB', 'PDT', 'NN', 'NNP', 'VBZ',...	1	
to	4	206700	2	0.026032	5.263587	TO	23	{'WDT', 'FW', 'RB', 'PDT', 'NN', 'NNP', 'VBZ',...	1	
a	5	189310	1	0.023842	5.390375	DT	21	{'RBR', 'PRP', 'FW', 'RB', 'NN', 'NNP', 'VBZ',...	1	

Get Lexicon

In [15]:
```
SALEX = pd.read_csv('../salex/salex_nrc.csv').set_index('term_str')
SALEX.columns = [col.replace('nrc_','') for col in SALEX.columns]
```

In [16]:
```
SALEX
```

Out[16]:

term_str	anger	anticipation	disgust	fear	joy	negative	positive	sadness	surprise	trust
abandon	0	0	0	1	0	1	0	1	0	0
abandoned	1	0	0	1	0	1	0	1	0	0
abandonment	1	0	0	1	0	1	0	1	1	0
abduction	0	0	0	1	0	1	0	1	1	0
aberration	0	0	1	0	0	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
young	0	1	0	0	1	0	1	0	1	0

term_str	anger	anticipation	disgust	fear	joy	negative	positive	sadness	surprise	trust
youth	1	1	0	1	1	0	1	0	1	0
zeal	0	1	0	0	1	0	1	0	1	1
zealous	0	0	0	0	1	0	1	0	0	1
zest	0	1	0	0	1	0	1	0	0	1

3688 rows × 11 columns

In [17]:
```
BOW = create_bow(CORPUS, CHAPS)
```

In [18]:
```
BOW
```

Out[18]:

book_id	chap_id	term_str	n
70	1	1835	1
		1910	1
		a	2
		alphabet	1
		as	2
...	...	...	...
62739	6	will	1
		with	1
		would	1
		year	1
		you	1

2307752 rows × 1 columns

Create COMBO table

In [19]:
```
COMBO = CORPUS.join(LIB).join(SALEX, on='term_str').join(BOW, on=OHCO[:2] + ['te
COMBO = COMBO.drop(['n'], axis=1)
COMBO = COMBO.sort_index()
```

In [20]:
```
COMBO
```

Out[20]:

book_id	chap_id	para_num	sent_num	token_num	pos_tuple	pos	token_str	term_str

book_id	chap_id	para_num	sent_num	token_num	pos_tuple	pos	token_str	term_str	
70	**1**	**1**	**0**	**0**					
					('By', 'IN')	IN	By	by	Tv
				1	('Mark', 'NNP')	NNP	Mark	mark	Tv
				2	('Twain', 'NNP')	NNP	Twain	twain	Tv
			2	**0**	**0**				
					('(Samuel', 'JJ')	JJ	(Samuel	samuel	Tv
				1	('Langhorne', 'NNP')	NNP	Langhorne	langhorne	Tv
...	**...**	**...**	**...**	**...**	...	...	...	...	
62739	**6**	**13**	**0**	**8**					
					("Leopold's", 'NNP')	NNP	Leopold's	leopolds	kin
				9	('Soliloquy,', 'NNP')	NNP	Soliloquy,	soliloquy	kin
				10	('by', 'IN')	IN	by	by	kin
				11	('Mark', 'NNP')	NNP	Mark	mark	kin

book_id	chap_id	para_num	sent_num	token_num	pos_tuple	pos	token_str	term_str	
				12	('Twain', 'NNP')	NNP	Twain	twain	kin

7940320 rows × 25 columns

Sentiment by Book

In [21]:
```
books = COMBO.groupby(OHCO[:1])[emo_cols].mean().join(LIB[['label', 'type']])
```

In [22]:
```
books.style.background_gradient(cmap='GnBu', axis=None)
```

Out[22]:

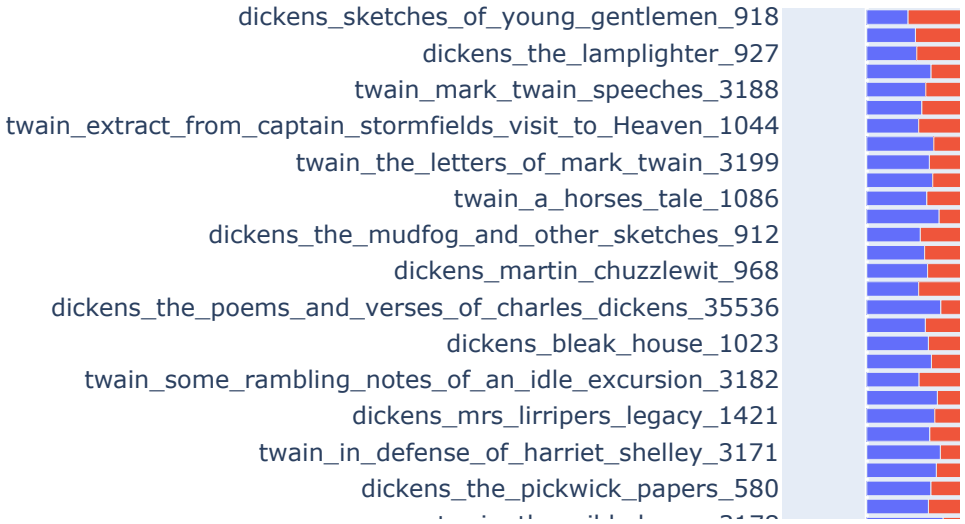
book_id	anger	anticipation	disgust	fear	joy	sadness	surprise	trust	
70	0.218018	0.293178	0.162472	0.288144	0.355147	0.270266	0.182086	0.449922	
74	0.270876	0.273164	0.230840	0.324640	0.304049	0.335392	0.179364	0.328300	-
76	0.290509	0.304715	0.224007	0.322215	0.380894	0.294832	0.225242	0.407453	
86	0.245507	0.278733	0.168067	0.315837	0.321138	0.272786	0.195217	0.427020	
91	0.251055	0.299578	0.196203	0.281294	0.355134	0.288326	0.194093	0.429677	
93	0.270936	0.261084	0.252874	0.343186	0.314450	0.344828	0.170772	0.399836	-
98	0.261961	0.280253	0.179526	0.333255	0.307810	0.295380	0.177181	0.398687	
102	0.283348	0.308050	0.189189	0.338274	0.340308	0.325196	0.210695	0.380994	-
119	0.219272	0.307659	0.153502	0.279690	0.367530	0.283186	0.202229	0.405878	
142	0.213292	0.326036	0.155121	0.277717	0.400625	0.284128	0.183894	0.426114	
245	0.230486	0.278304	0.164843	0.315181	0.315796	0.287892	0.185618	0.393362	
564	0.205431	0.291354	0.160950	0.259557	0.368703	0.250447	0.189532	0.434262	
580	0.200398	0.265022	0.163345	0.255339	0.310942	0.238316	0.176851	0.458030	
588	0.187770	0.295115	0.148887	0.273513	0.365238	0.248588	0.172483	0.419741	
644	0.207738	0.251958	0.185168	0.322893	0.326578	0.337632	0.160755	0.347766	
650	0.239856	0.283138	0.192741	0.297115	0.344905	0.314022	0.173805	0.351668	-
653	0.205102	0.304008	0.151484	0.262884	0.407080	0.266007	0.187923	0.417491	
675	0.232051	0.270353	0.173878	0.296795	0.334615	0.299038	0.148718	0.383173	

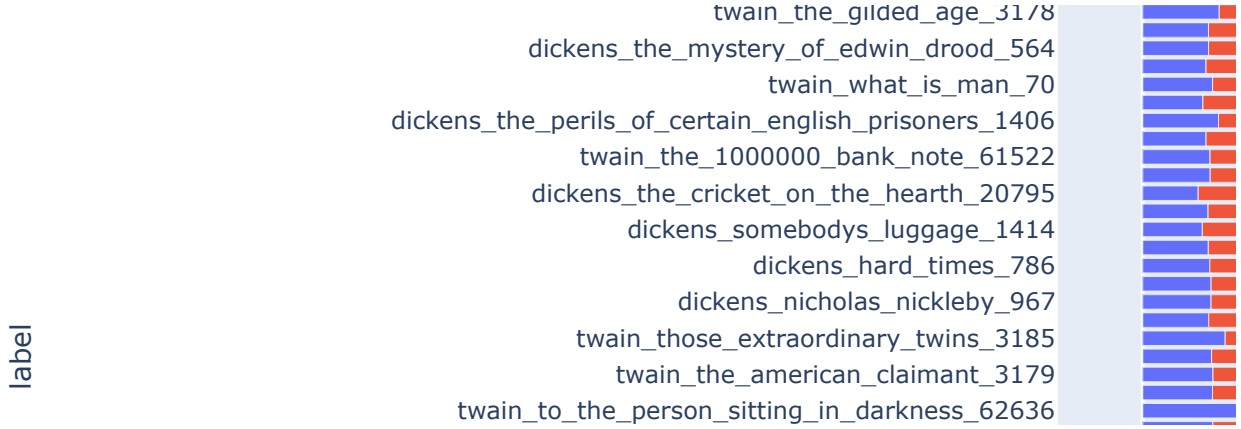
book_id	anger	anticipation	disgust	fear	joy	sadness	surprise	trust
676	0.171991	0.289137	0.107029	0.228967	0.425985	0.200213	0.179446	0.498935
699	0.313401	0.271915	0.223883	0.351968	0.293852	0.289253	0.158691	0.394870
700	0.192560	0.319235	0.150475	0.250349	0.394891	0.269961	0.176438	0.424274
730	0.241240	0.279724	0.204232	0.297343	0.311220	0.277264	0.180512	0.375492
766	0.182965	0.306261	0.140980	0.237849	0.398976	0.259059	0.179921	0.464779
786	0.209346	0.290654	0.173832	0.257009	0.344626	0.272897	0.191589	0.439486
807	0.248434	0.258873	0.189979	0.300626	0.313152	0.323591	0.187891	0.436326
809	0.197531	0.315376	0.158249	0.210999	0.402918	0.223345	0.166105	0.398429
810	0.197461	0.272214	0.179126	0.270804	0.332863	0.263752	0.184767	0.506347
821	0.203769	0.307539	0.162035	0.260036	0.387920	0.265758	0.183515	0.419755
824	0.151991	0.332917	0.126630	0.208787	0.426505	0.214324	0.181461	0.512413
872	0.243876	0.278638	0.179395	0.286383	0.331952	0.258646	0.172010	0.408141
882	0.217789	0.309579	0.175582	0.255020	0.359517	0.275902	0.183490	0.390972
883	0.214846	0.289870	0.196620	0.252888	0.365953	0.248649	0.190633	0.427678
888	0.236819	0.279170	0.178479	0.300778	0.312446	0.309853	0.197061	0.378997
912	0.167232	0.278236	0.162870	0.231701	0.304896	0.214251	0.168202	0.457586
914	0.224947	0.282716	0.195355	0.290969	0.340839	0.284367	0.176020	0.389531
916	0.180591	0.315612	0.153586	0.214346	0.423629	0.225316	0.171308	0.447257
917	0.227854	0.263985	0.187443	0.296693	0.314605	0.269426	0.172587	0.406859
918	0.128987	0.438696	0.094968	0.165840	0.508859	0.143161	0.356485	0.442240
922	0.226018	0.345598	0.169514	0.249671	0.411301	0.253614	0.170828	0.416557
927	0.156118	0.310127	0.111814	0.189873	0.352321	0.206751	0.229958	0.512658
967	0.213126	0.291593	0.178251	0.256431	0.357940	0.258443	0.194347	0.426731
968	0.189850	0.288757	0.155923	0.233625	0.367826	0.226180	0.188389	0.456884
1023	0.192444	0.298300	0.153908	0.235951	0.364768	0.241490	0.185201	0.469252
1044	0.162044	0.383942	0.140146	0.192701	0.470073	0.229197	0.214599	0.494891
1086	0.187443	0.316652	0.158326	0.236579	0.414923	0.256597	0.194722	0.422202
1289	0.221018	0.254379	0.201835	0.326939	0.247706	0.295246	0.166806	0.386989
1394	0.205917	0.294675	0.165680	0.220118	0.396450	0.244970	0.190533	0.469822
1400	0.232910	0.273758	0.194267	0.289300	0.327208	0.285834	0.169484	0.393153
1406	0.236449	0.277570	0.148598	0.271028	0.338318	0.273832	0.183178	0.414019
1407	0.200803	0.377510	0.105756	0.191432	0.465863	0.202142	0.265060	0.481928

book_id	anger	anticipation	disgust	fear	joy	sadness	surprise	trust
1413	0.227513	0.266314	0.236332	0.245150	0.315697	0.296296	0.144621	0.403880
1414	0.185866	0.299129	0.129719	0.228461	0.369797	0.299129	0.169409	0.420136
1415	0.209302	0.377907	0.133721	0.168605	0.494186	0.168605	0.226744	0.395349
1416	0.161981	0.336011	0.140562	0.218206	0.467202	0.218206	0.190094	0.469880
1421	0.212299	0.300146	0.178624	0.222548	0.379209	0.260615	0.181552	0.465593
1435	0.330178	0.264477	0.246102	0.376949	0.290089	0.299555	0.180401	0.367483
1467	0.196781	0.310900	0.196781	0.216533	0.428676	0.255304	0.193855	0.412582
1837	0.235863	0.259962	0.218216	0.310816	0.306641	0.298672	0.175142	0.390133
2324	0.195901	0.315539	0.154909	0.267874	0.370353	0.296949	0.170639	0.407531
2874	0.247254	0.297983	0.159177	0.322349	0.349111	0.271021	0.182544	0.399641
2875	0.279614	0.304604	0.161354	0.374065	0.321921	0.302243	0.162731	0.377410
2895	0.237565	0.297570	0.173787	0.316870	0.348188	0.273006	0.192824	0.410299
3171	0.229814	0.313221	0.197870	0.253771	0.402839	0.266193	0.174800	0.394854
3172	0.210526	0.298246	0.100877	0.250000	0.364035	0.289474	0.241228	0.412281
3173	0.201908	0.271860	0.174881	0.241653	0.365660	0.225755	0.240064	0.492846
3176	0.231709	0.292387	0.174175	0.291339	0.361009	0.285228	0.175746	0.393924
3177	0.264843	0.278414	0.183206	0.327290	0.302481	0.293681	0.180874	0.390373
3178	0.238616	0.323363	0.158949	0.257118	0.364778	0.273799	0.194037	0.448567
3179	0.219057	0.283045	0.200374	0.269967	0.365950	0.279542	0.197571	0.430406
3180	0.245833	0.313333	0.181667	0.320000	0.355000	0.306667	0.210000	0.386667
3181	0.244792	0.281250	0.127604	0.304688	0.286458	0.260417	0.221354	0.481771
3182	0.163432	0.317671	0.148110	0.249234	0.411645	0.255363	0.216547	0.450460
3183	0.252390	0.214149	0.254302	0.296367	0.258126	0.282983	0.133843	0.399618
3184	0.218612	0.308715	0.162482	0.265879	0.374200	0.305761	0.204333	0.413589
3185	0.257292	0.259536	0.141361	0.299925	0.275991	0.250561	0.166043	0.445774
3186	0.257972	0.293802	0.206019	0.336797	0.361161	0.288785	0.198495	0.359011
3188	0.184236	0.319212	0.133990	0.223251	0.400985	0.224433	0.186010	0.519015
3189	0.253662	0.267529	0.199283	0.322686	0.310221	0.307884	0.171705	0.382830

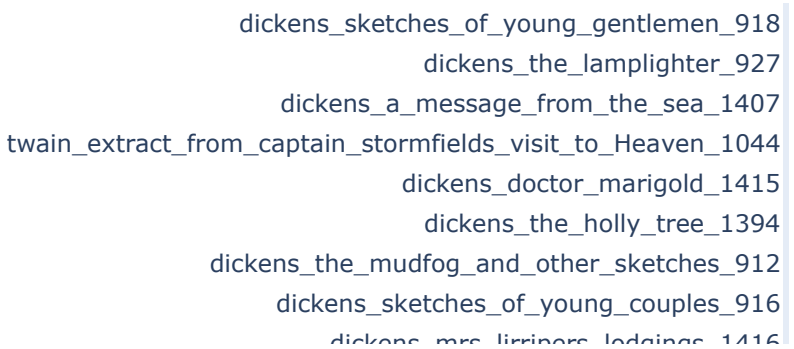
book_id	anger	anticipation	disgust	fear	joy	sadness	surprise	trust
3190	0.217327	0.259912	0.220264	0.212922	0.348018	0.193833	0.152717	0.436123
3191	0.304450	0.259953	0.217799	0.346604	0.271663	0.278689	0.142857	0.377049
3192	0.277589	0.280136	0.231749	0.345501	0.295416	0.285229	0.180815	0.376910
3199	0.195551	0.340659	0.147474	0.235513	0.425451	0.247346	0.205355	0.457096
3250	0.205882	0.278075	0.152406	0.272727	0.377005	0.286096	0.229947	0.390374
3251	0.241768	0.289614	0.157473	0.302420	0.333521	0.284548	0.179567	0.433014
19337	0.211608	0.311971	0.178356	0.266626	0.417170	0.265417	0.206771	0.377267
19484	0.297638	0.258268	0.193701	0.346457	0.297638	0.348031	0.196850	0.377953
19987	0.204595	0.326821	0.151922	0.262426	0.377931	0.283214	0.202563	0.432948
20795	0.172968	0.289698	0.201796	0.199905	0.436673	0.206994	0.189509	0.378544
27924	0.229795	0.266667	0.165363	0.300559	0.334078	0.290503	0.168715	0.384358
33077	0.220524	0.292576	0.155022	0.334061	0.312227	0.222707	0.141921	0.495633
35536	0.231201	0.305275	0.169473	0.241302	0.432099	0.261504	0.198653	0.406285
60900	0.232446	0.251816	0.167554	0.331235	0.291525	0.280387	0.175787	0.389346
61522	0.210055	0.312593	0.137382	0.293430	0.376804	0.263813	0.186909	0.444251
62636	0.298701	0.238961	0.140260	0.303896	0.303896	0.272727	0.145455	0.464935
62739	0.301115	0.257745	0.231722	0.395291	0.245353	0.322181	0.146221	0.348203

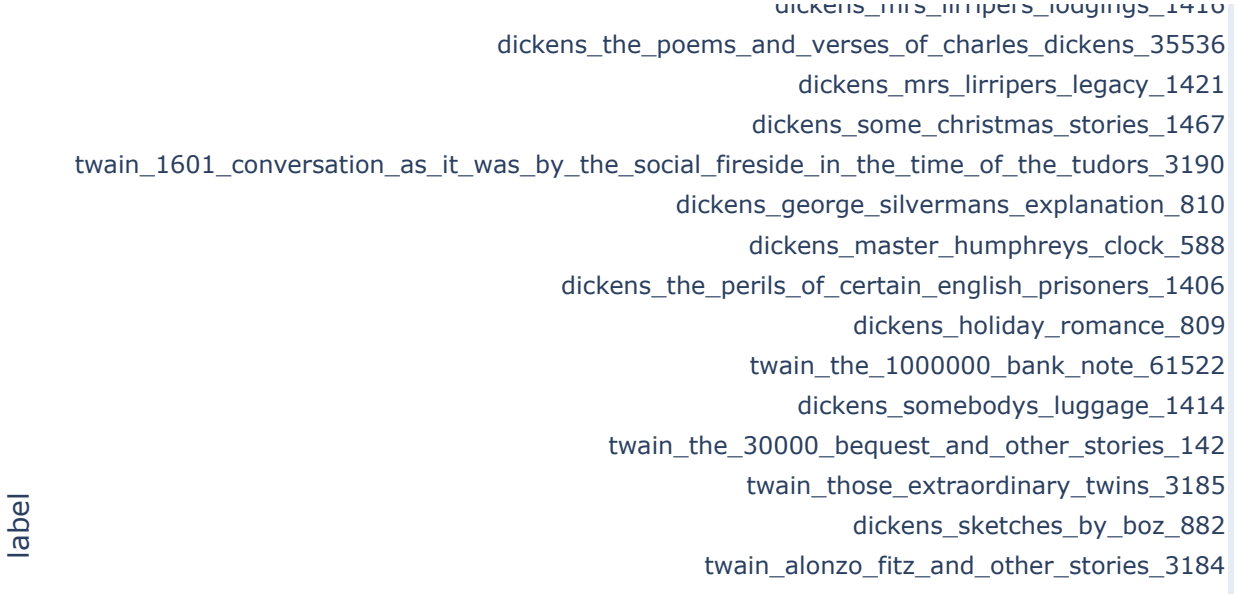
In [23]:
```
px.bar(books.reset_index().sort_values('polarity'), emo_cols, 'label', orientati
```



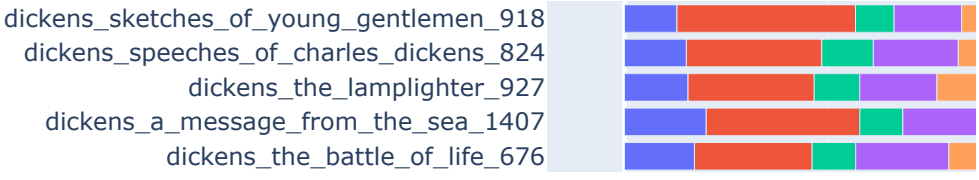


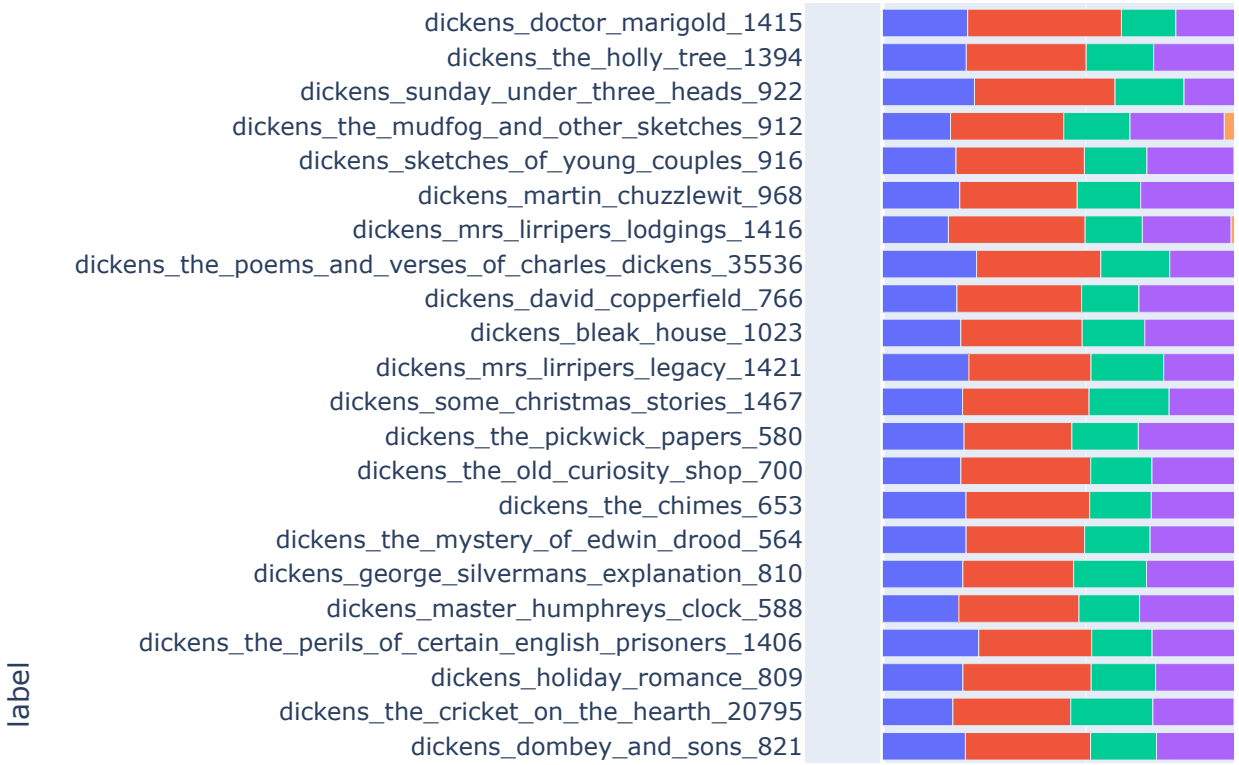
In [24]:
```
px.bar(books.loc[books.type == 'stories'].reset_index().sort_values('polarity'),
```





In [25]:
```
px.bar(books.loc[books.label.str.contains('dickens', case = False)].reset_index(
```





In [26]:
```
px.bar(books.loc[books.label.str.contains('twain', case = False)].reset_index().
```

label

twain_mark_twain_speeches_3188
twain_extract_from_captain_stormfields_visit_to_Heaven_1044
twain_the_letters_of_mark_twain_3199
twain_a_horses_tale_1086
twain_essays_on_paul_bourget_3173
twain_some_rambling_notes_of_an_idle_excursion_3182
twain_the_treaty_with_china_its_provisions_explained_33077
twain_in_defense_of_harriet_shelley_3171
twain_1601_conversation_as_it_was_by_the_social_fireside_in_the_time_of_the_tudors_3190
twain_the_gilded_age_3178
twain_what_is_man_70
twain_the_1000000_bank_note_61522
twain_fenimore_coopers_literary_offences_3172
twain_chapters_from_my_autobiography_19987
twain_the_30000_bequest_and_other_stories_142
twain_how_to_tell_a_story_and_other_essays_3250
twain_those_extraordinary_twins_3185
twain_the_american_claimant_3179
twain_to_the_person_sitting_in_darkness_62636
twain_alonzo_fitz_and_other_stories_3184
twain_a_tramp_abroad_119
twain_a_connecticut_yankee_in_king_arthurs_court_86
twain_the_innocents_abroad_3176
twain_the_man_that_corrupted_hadleyburg_and_other_stories_3251

In [77]:

```
books.loc[books.polarity <= 0].shape[0] / books.shape[0]
```

Out[77]:
```
0.17894736842105263
```

In [79]:
```
books.loc[books.polarity <= 0].type.value_counts()
```

Out[79]:
```
stories        9
novel          4
non-fiction    4
Name: type, dtype: int64
```

In [86]:
```
# number of books with neutral / negative polarity by dickens
books.loc[(books.polarity <= 0) & (books.label.str.contains('dickens', case = Fa
```

Out[86]:
```
5
```

In [87]:
```
# number of books with neutral / negative polarity by twain
books.loc[(books.polarity <= 0) & (books.label.str.contains('twain', case = Fals
```

Out[87]:
```
12
```

Compare Novels

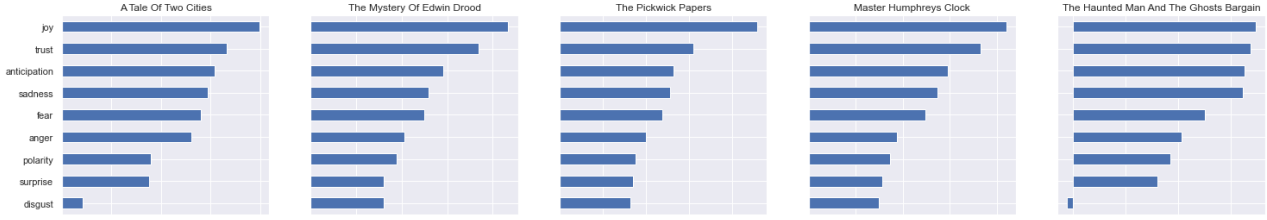
Get Novels

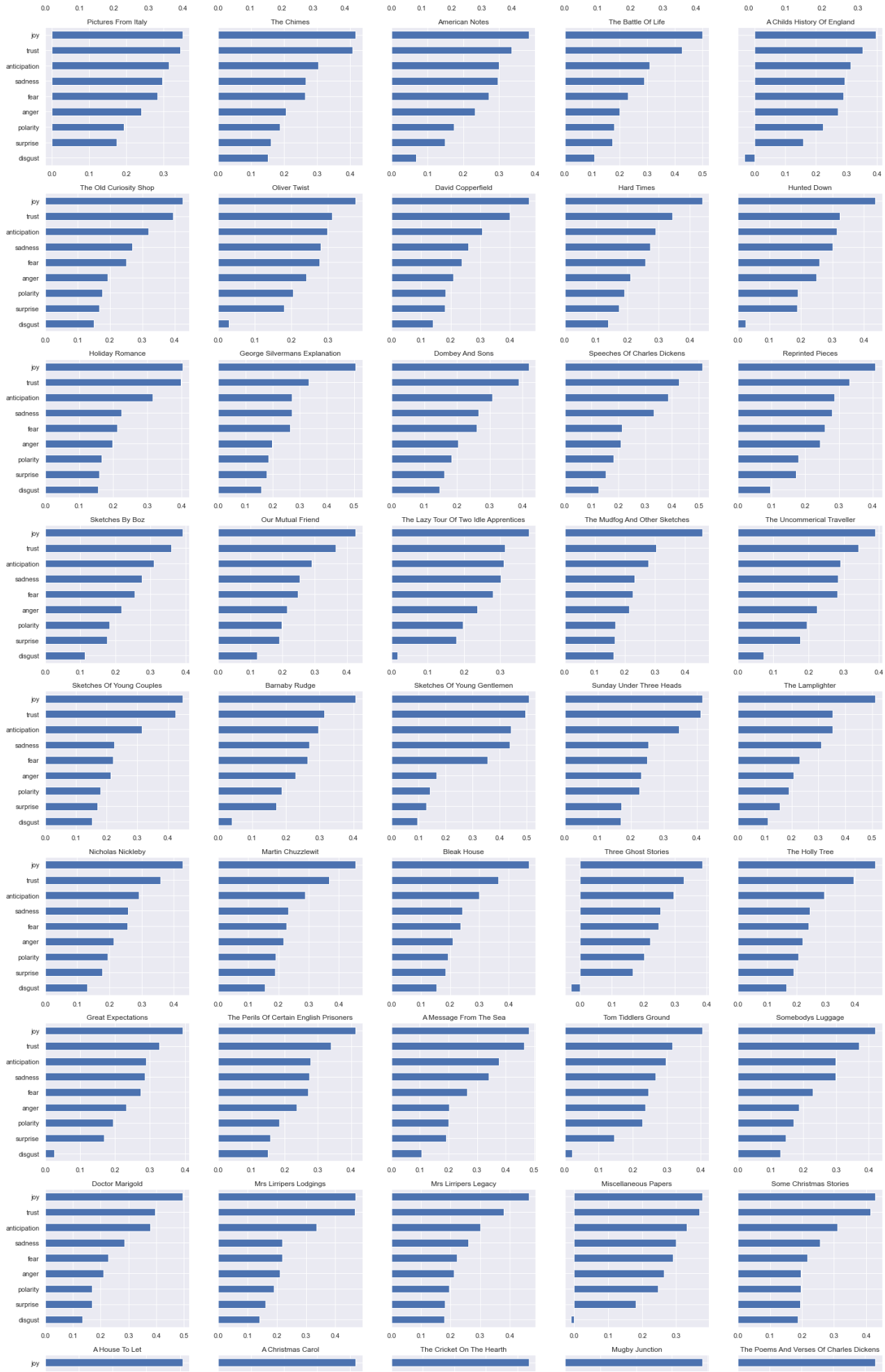
In [27]:
```
LIB.author.value_counts()
```

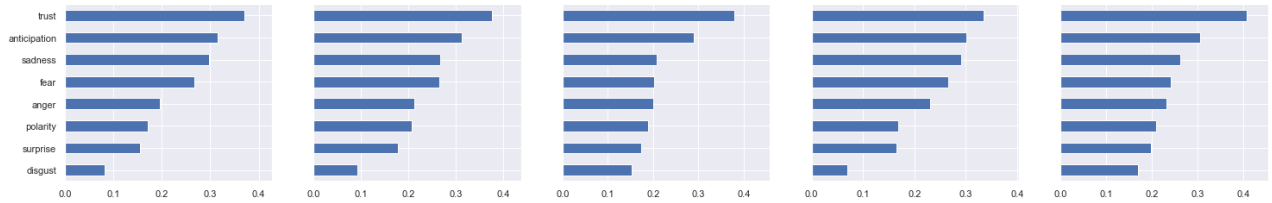
Out[27]:
```
dickens    50
twain      45
Name: author, dtype: int64
```

In [28]:
```
dickens_books = LIB.loc[LIB.author == 'dickens'].index.values

twain_books = LIB.loc[LIB.author == 'twain'].index.values
```

In [29]:
```
nrows = int(LIB.loc[LIB.author == 'dickens'].shape[0] / 5)
ncols = 5

fig, axes = plt.subplots(nrows = nrows, ncols = ncols, sharey = True, figsize =

for row in range(nrows):
    for col in range(ncols):
        index = ncols * row + col
        books.loc[dickens_books[index], emo_cols].sort_values().plot.barh(ax = a
```

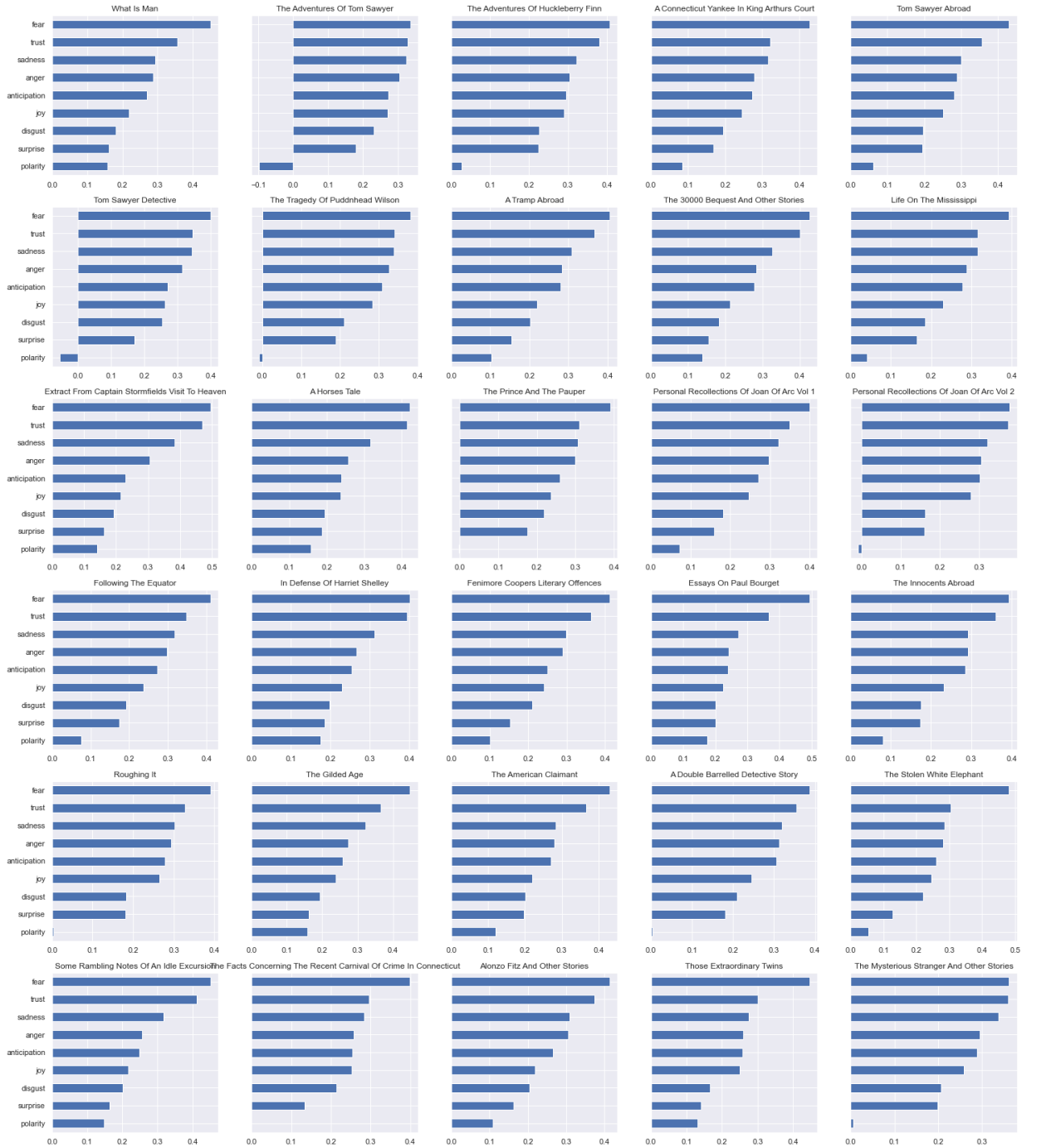



Pictures From Italy
The Chimes
American Notes
The Battle Of Life
A Childs History Of England
The Old Curiosity Shop
Oliver Twist
David Copperfield
Hard Times
Hunted Down
Holiday Romance
George Silvermans Explanation
Dombey And Sons
Speeches Of Charles Dickens
Reprinted Pieces
Sketches By Boz
Our Mutual Friend
The Lazy Tour Of Two Idle Apprentices
The Mudfog And Other Sketches
The Uncommerical Traveller
Sketches Of Young Couples
Barnaby Rudge
Sketches Of Young Gentlemen
Sunday Under Three Heads
The Lamplighter
Nicholas Nickleby
Martin Chuzzlewit
Bleak House
Three Ghost Stories
The Holly Tree
Great Expectations
The Perils Of Certain English Prisoners
A Message From The Sea
Tom Tiddlers Ground
Somebodys Luggage
Doctor Marigold
Mrs Lirripers Lodgings
Mrs Lirripers Legacy
Miscellaneous Papers
Some Christmas Stories
A House To Let
A Christmas Carol
The Cricket On The Hearth
Mugby Junction
The Poems And Verses Of Charles Dickens
joy
trust
anticipation
sadness
fear
anger
polarity
surprise
disgust



In [30]:

```
nrows = int(LIB.loc[LIB.author == 'twain'].shape[0] / 5)
ncols = 5

fig, axes = plt.subplots(nrows = nrows, ncols = ncols, sharey = True, figsize =

for row in range(nrows):
    for col in range(ncols):
        index = ncols * row + col
        books.loc[twain_books[index], emo_cols].sort_values().plot.barh(ax = axe
```



In [31]:
```
# Just for convenience
class Novels(): pass
novels = Novels()
for idx in LIB.index:
    label = LIB.loc[idx].label
    label = label.replace(' ', '_')
    label = label.replace('—','_')
    setattr(novels, label, idx)
```

VOCAB Table

In [32]:
```
# dickens books
two_cities = novels.dickens_a_tale_of_two_cities_98
great_expectations = novels.dickens_great_expectations_1400
copperfield = novels.dickens_david_copperfield_766
twist = novels.dickens_oliver_twist_730
bleak = novels.dickens_bleak_house_1023
mutual = novels.dickens_our_mutual_friend_883

dickens_works = [(LIB.loc[two_cities, "title"], two_cities),
                 (LIB.loc[great_expectations, "title"], great_expectations),
                 (LIB.loc[copperfield, "title"], copperfield),
                 (LIB.loc[twist, "title"], twist),
                 (LIB.loc[bleak, "title"], bleak),
                 (LIB.loc[mutual, "title"], mutual)]
```

In [33]:
```
# twain books
rough = novels.twain_roughing_it_3177
gilded = novels.twain_the_gilded_age_3178
sawyer = novels.twain_the_adventures_of_tom_sawyer_74
huck = novels.twain_the_adventures_of_huckleberry_finn_76
yankee = novels.twain_a_connecticut_yankee_in_king_arthurs_court_86
puddnhead = novels.twain_the_tragedy_of_puddnhead_wilson_102
```

```
twain_works = [(LIB.loc[rough, "title"], rough),
               (LIB.loc[gilded, "title"], gilded),
               (LIB.loc[sawyer, "title"], sawyer),
               (LIB.loc[huck, "title"], huck),
               (LIB.loc[yankee, "title"], yankee),
               (LIB.loc[puddnhead, "title"], puddnhead)
]
```

Plot Sentiments

In [34]:
```
def plot_sentiments(df, title, emo='polarity'):
    FIG = dict(figsize=(25, 5), legend=True, fontsize=14, title = title)
    df[emo].plot(**FIG)
```

In [35]:
```
chaps_df = COMBO.groupby(OHCO[:2])[emo_cols].mean()
```

In [36]:
```
chaps_df = chaps_df.join(LIB)
```

In [37]:
```
chaps_df.head()
```

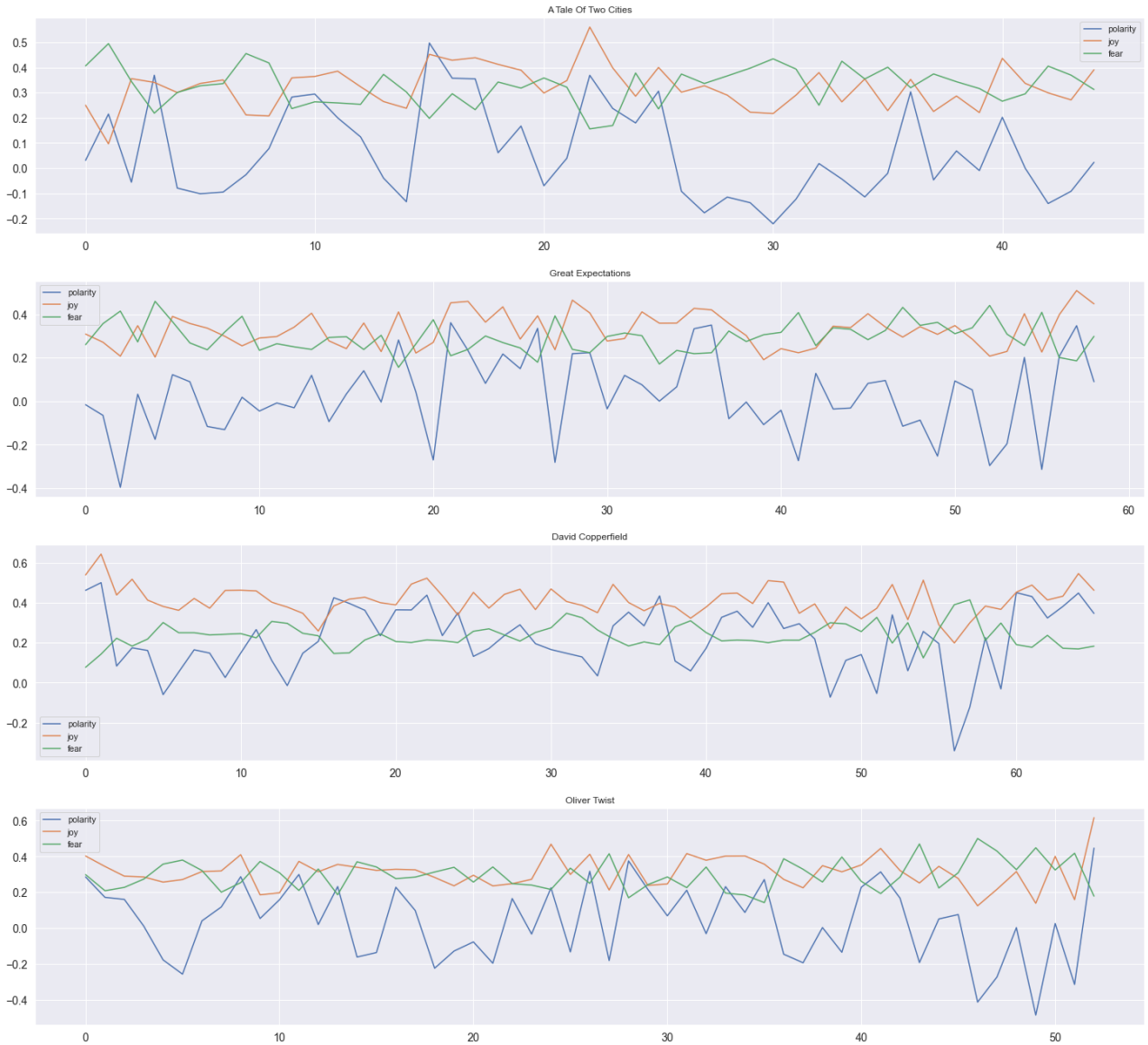
Out[37]:

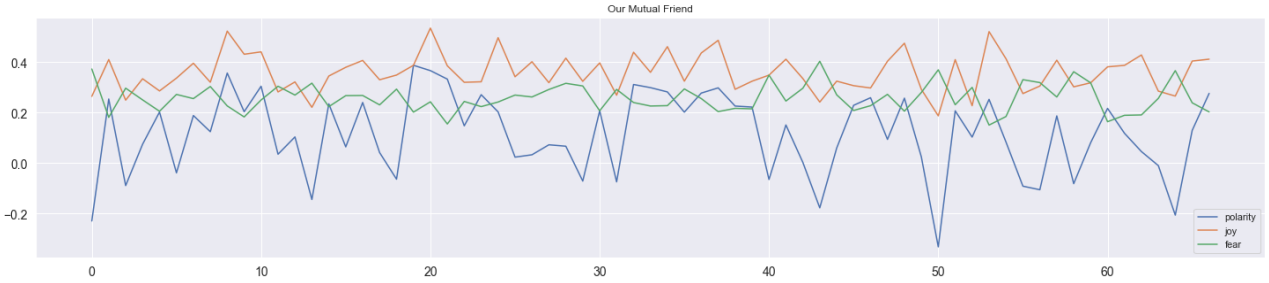
		anger	anticipation	disgust	fear	joy	sadness	surprise	
book_id	chap_id								
70	1								
		0.400000	0.600000	0.200000	0.400000	0.600000	0.400000	0.600000	0.6
	2	0.208876	0.283432	0.174556	0.284615	0.357396	0.270414	0.146154	0.4
	3	0.190647	0.320144	0.118705	0.287770	0.489209	0.334532	0.233813	0.4
	4	0.248756	0.318408	0.199005	0.373134	0.278607	0.318408	0.208955	0.3

book_id	chap_id	anger	anticipation	disgust	fear	joy	sadness	surprise	
5		0.175115	0.308756	0.129032	0.221198	0.368664	0.276498	0.239631	0.4

Dickens Books

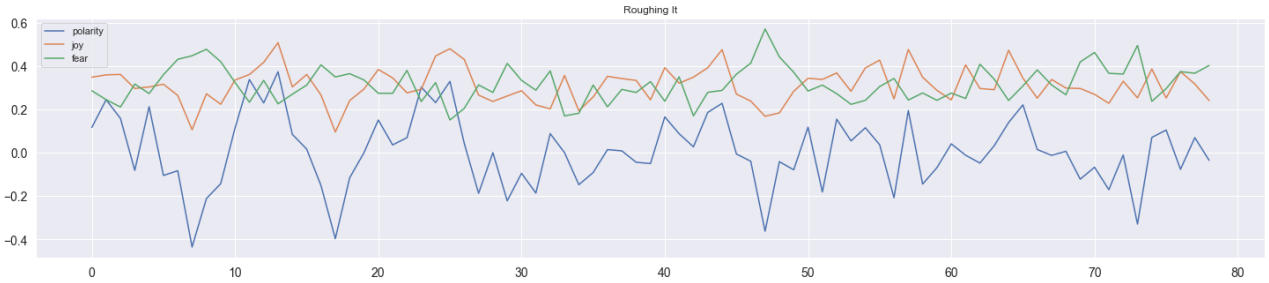
In [38]:
```
for col, book in enumerate(dickens_works):
    plot_sentiments(chaps_df.loc[book[1]].reset_index(), book[0].title(), ['pola
```

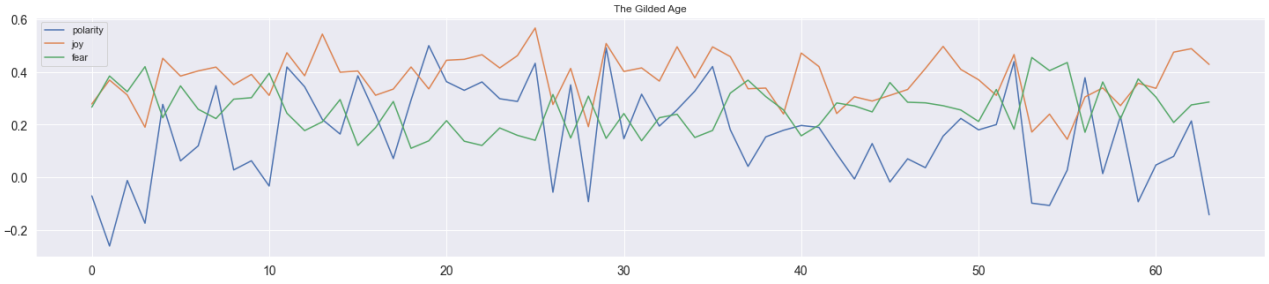


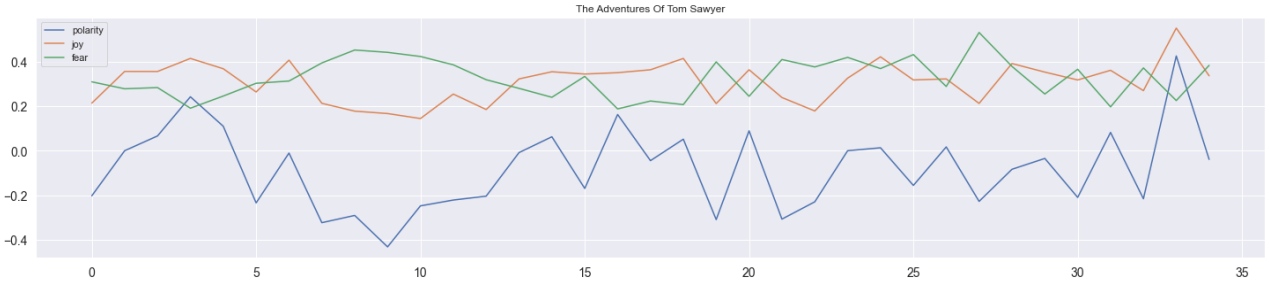


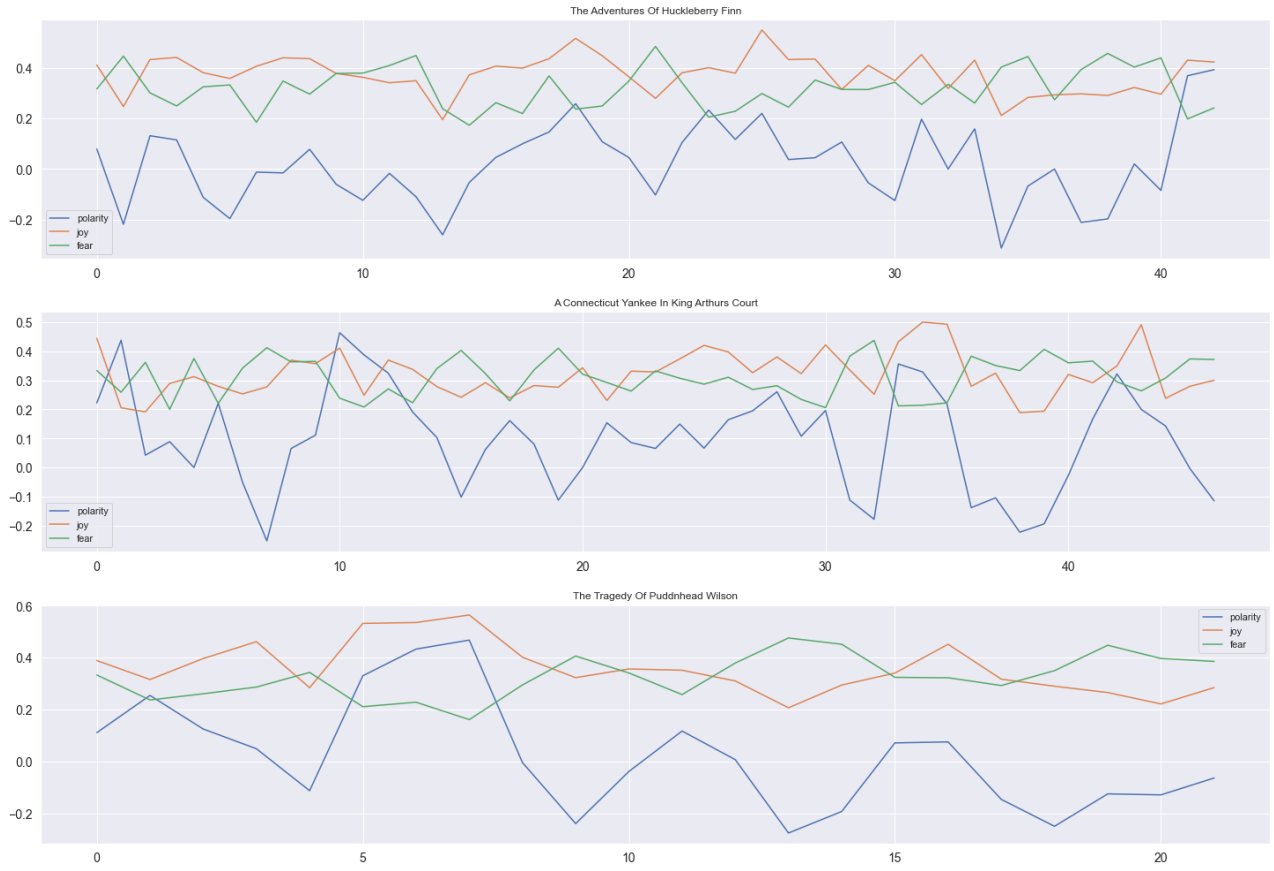
Twain Books

In [39]:
```
for col, book in enumerate(twain_works):
    plot_sentiments(chaps_df.loc[book[1]].reset_index(), book[0].title(), ['pola
```









Cose Read Sentiment in Texts

In [40]:
```
COMBO['html'] = COMBO.fillna(0).apply(lambda x: f"<spdan class='sent{int(np.sign
```

In [41]:
```
COMBO.html.head()
```

Out[41]:
```
book_id  chap_id  para_num  sent_num  token_num
70       1        1         0         0                  <spdan class='sent0'>b
y</span>
                                      1                <spdan class='sent0'>mar
k</span>
                                      2               <spdan class='sent0'>twai
n</span>
                   2         0         0              <spdan class='sent0'>samue
l</span>
                                      1           <spdan class='sent0'>langhorn
e</span>
Name: html, dtype: object
```

In [42]:
```
SENTENCES = COMBO.groupby(OHCO[:-1])[emo_cols].mean() #.term_str.count().to_fram
```

In [43]:
```
SENTENCES['html_str'] = COMBO.groupby(OHCO[:-1]).html.apply(lambda x: x.str.cat(
```

In [44]:
```
def sample_sentences(df, sample_size=10, emo='polarity'):
```

```
    rows = []
    sample = df.dropna().sample(sample_size).index
    for idx in sample:
        valence = round(df.loc[idx, emo], 4)
        id_label = ' '.join([str(i) for i in idx]).upper()
        t = 0
        if valence > t: color = '#ccffcc'
        elif valence < t: color = '#ffcccc'
        else: color = '#f2f2f2'
        z = 0
        rows.append("""
        <tr style="background-color:{0};padding:.5rem 1rem;font-size:110%;">
            <td style="width:20%;">{1}</td>
            <td style="width:70%;">{2}</td>
            <td>{3}</td>
        </tr>
        """.format(color, id_label, df.loc[idx, 'html_str'], valence))

    css = """
    #sample1 td {font-size:110%;vertical-align:top;text-align:left;}
    #sample1 th {font-size:120%;vertical-align:top;text-align:left;}
    .sent-1 {color:red;font-weight:bold;}
    .sent1 {color:green;font-weight:bold;}
    """
    display(HTML(f'<style>{css}</style>'))
    display(HTML('<table id="sample1"><tr><th>Sentence</th><th>ID</th><th>Sentim
```

In [45]:

```
sample_sentences(SENTENCES.loc[great_expectations])
```

Sentence	ID	Sentiment
11 123 1	he seemed to have no **strength and he never once hit me hard and he was always knocked down but he would be up again in a moment sponging himself or drinking out of the water bottle with the greatest satisfaction in seconding himself according to form and then came at me with an air and a show that made me believe he really was going to do for me at last**	0.0
40 54 2	i want to know how you are to be kept out of **danger how long you are going to stay what projects you have**	-1.0
45 6 1	the little **servant happening to be entering the fortress with two hot rolls i passed through the postern and crossed the drawbridge in her company and so came without announcement into the presence of wemmick as he was making tea for himself and the aged**	0.0
55 26 0	herbert was highly **delighted when we shook hands on this arrangement and said he could now take courage to tell me that he believed he must go away at the end of the week**	1.0
4 20 0	joes station and influence were something feebler if possible when there was company than when there was none	0.0

2 60 2	i had begun by asking questions and i was going to **rob mrs joe**	-1.0
2 22 1	after that he sat feeling his right side flaxen curls and whisker and following mrs joe about with his blue eyes as his manner always was at squally times	0.0
5 20 0	the interest of the impending pursuit not only absorbed the **general attention but even made my sister liberal**	0.3333
26 46 0	i told him i had come up again to say how sorry i was that anything disagreeable should have occurred and that i hoped he would not **blame me much**	-1.0
22 42 1	miss havisham you must know was a spoilt **child**	1.0

In [46]:
```
sample_sentences(SENTENCES.loc[bleak])
```

Sentence	ID	Sentiment
50 81 0	nothing could be more acceptable to the little circle than this call upon **young woolwich who immediately fetches his fife and performs the stirring melody during which performance mr bucket much enlivened beats time and never fails to come in sharp with the burden british gra a anadeers**	0.5
30 70 6	i have told your ladyship that i should be placed in a very disagreeable situation if any **complaint was made and all is in strict confidence**	0.0
37 72 4	might it not prove a little **worse than she expected**	-1.0
29 69 1	i **hope your time is not so precious but that you will allow my lady and myself to offer you the hospitality of chesney wold for to night at least**	1.0
14 118 1	all the **love and duty i could ever have rendered to him is transferred to you**	1.0
5 5 0	and mr jellyby **sir**	1.0
29 41 0	i am **happy lady dedlock that you say so and i need not comment on the value to me of your kind opinion of her**	1.0
14 26 1	said my **guardian**	1.0
46 94 4	thank you and **god bless you in her name**	1.0
15 99 0	he appears to be an **excellent master i observed**	1.0

In [47]:
```
sample_sentences(SENTENCES.loc[puddnhead])
```

Sentence	ID	Sentiment

4 26 0	deyll sell dese niggers to day fo stealin de **money den deyll buy some mo dat dont know de chillen so dats all right**	1.0
12 30 0	on the whole quite **fairly said luigi**	1.0
2 19 1	they **fell away from him as from something uncanny and went into privacy to discuss him**	-1.0
22 80 2	the percy driscoll estate was in such a **crippled shape when its owner died that it could pay only sixty per cent**	0.0
4 7 0	she put down the **child and made the change**	1.0
20 38 4	this charmed the doting mrs pratt who realized now as she had never done before she said what a sensitive and delicate nature her **darling had and how he adored his poor uncle**	0.5
14 59 1	but why cant she **pawn it or sell it**	-1.0
8 8 0	wilson had quickly chosen a position from which he could watch the girl without running much **risk of being seen by her and he remained there hoping she would raise her veil and betray her face**	-0.6667
22 13 6	this produced a strong sensation the last drowsy head in the court room roused up now and made preparation to listen	0.0
11 5 3	how hard the niggers **fate seems this morning yet until last night such a thought never entered my head**	-1.0

VADER

In [48]:
```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
```

In [49]:
```
analyser = SentimentIntensityAnalyzer()
```

In [50]:
```
SENTENCES['sent_str'] = COMBO.groupby(OHCO[:-1]).term_str.apply(lambda x: x.str.
vader_cols = [f"vader_{col}" for col in "neg neu pos compound".split()]
SENTENCES[vader_cols] = SENTENCES.sent_str.apply(analyser.polarity_scores).apply
```

In [51]:
```
SENTENCES.sample(10)
```

Out[51]:

				anger	anticipation	disgust	fear	joy	sa
book_id	chap_id	para_num	sent_num						
968	25	150	1						
				0.000000	0.000000	0.0	1.000000	0.000000	0.0

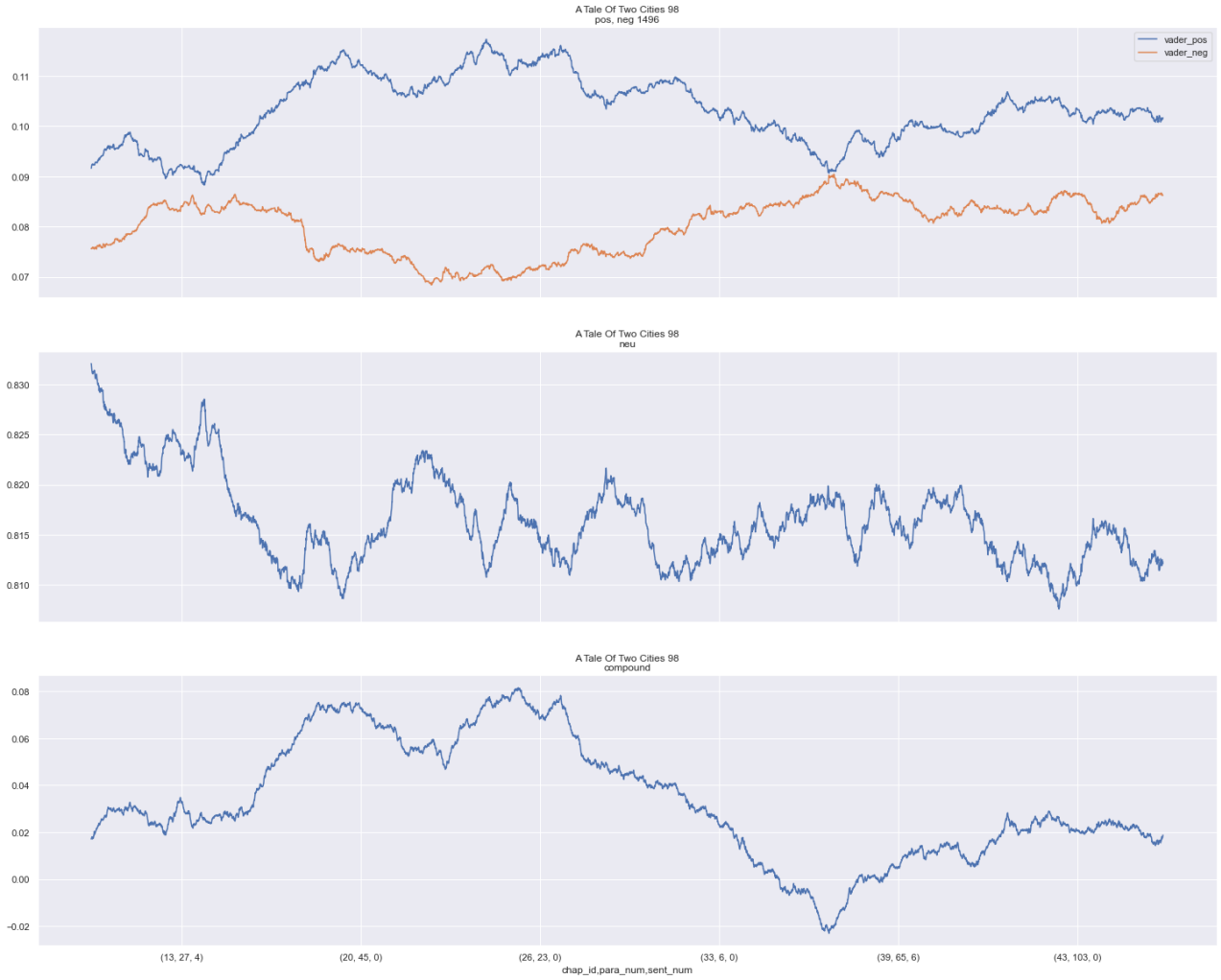
book_id	chap_id	para_num	sent_num	anger	anticipation	disgust	fear	joy	sa
1400	3	32	0	0.000000	0.000000	0.0	1.000000	1.000000	0.0
245	56	17	1	0.600000	0.400000	0.2	0.800000	0.000000	0.4
3176	26	5	6	0.222222	0.444444	0.0	0.444444	0.222222	0.3
1023	52	47	3	NaN	NaN	NaN	NaN	NaN	
564	17	142	1	NaN	NaN	NaN	NaN	NaN	
580	25	6	3	NaN	NaN	NaN	NaN	NaN	
882	54	159	7	0.250000	0.500000	0.0	0.750000	0.000000	0.2
61522	3	132	3	1.000000	0.000000	0.0	1.000000	0.000000	0.0
3199	9	77	0	NaN	NaN	NaN	NaN	NaN	

In [52]:
```
def vader_plot(novel_name):
    global SENTENCES
```

```
    X = SENTENCES.loc[novel_name]
    w = int(len(X)/5)
    fig, axes = plt.subplots(ncols=1, nrows=3, figsize=(25,20), sharex=True)
    X[['vader_pos','vader_neg']].rolling(w).mean().plot(ax=axes[0], title=f'{LIB
    X['vader_neu'].rolling(w).mean().plot(ax=axes[1], title=f'{LIB.loc[novel_nam
    X['vader_compound'].rolling(w).mean().plot(ax=axes[2], title=f'{LIB.loc[nove
```

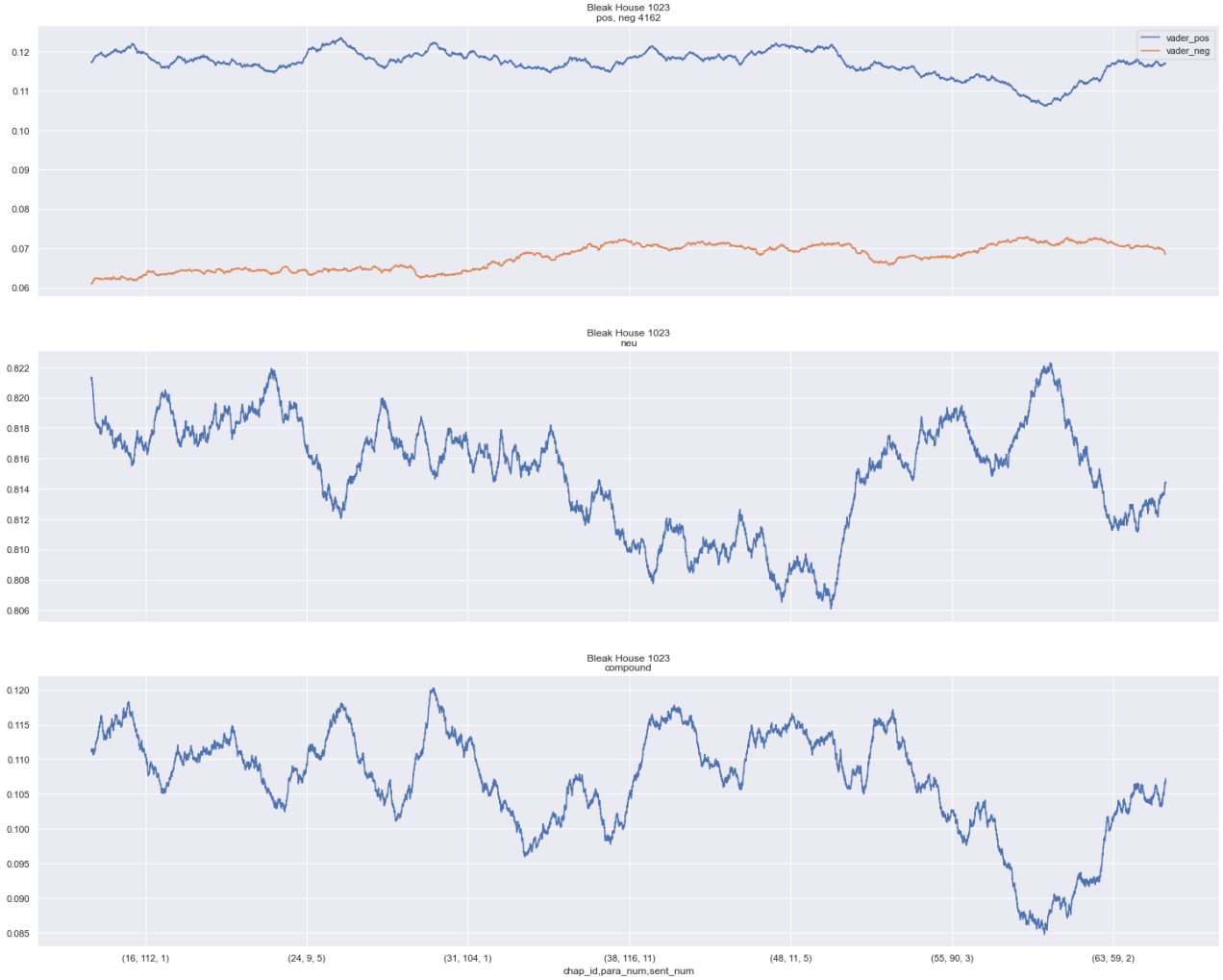
In [53]:

```
vader_plot(two_cities)
```

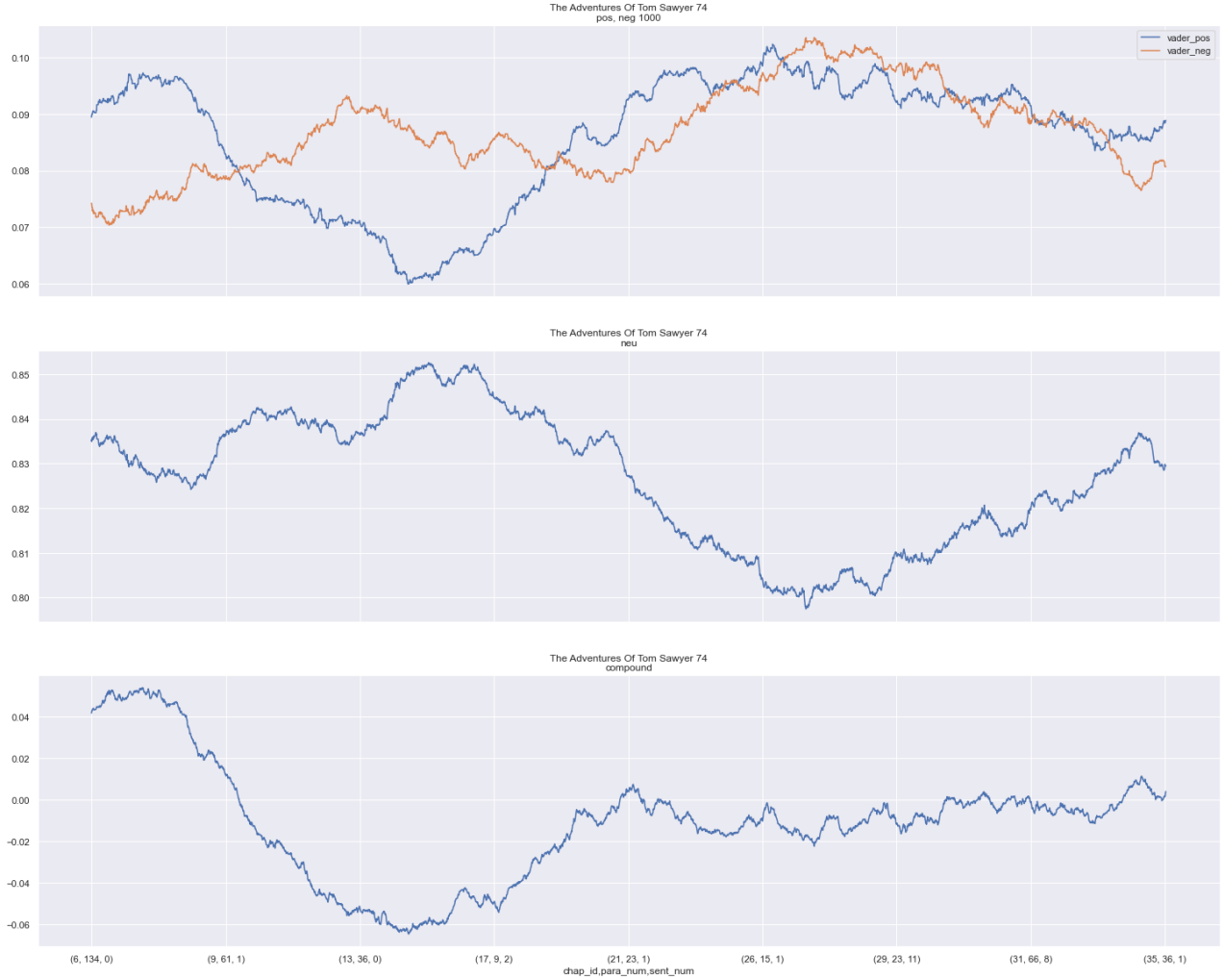


In [54]:

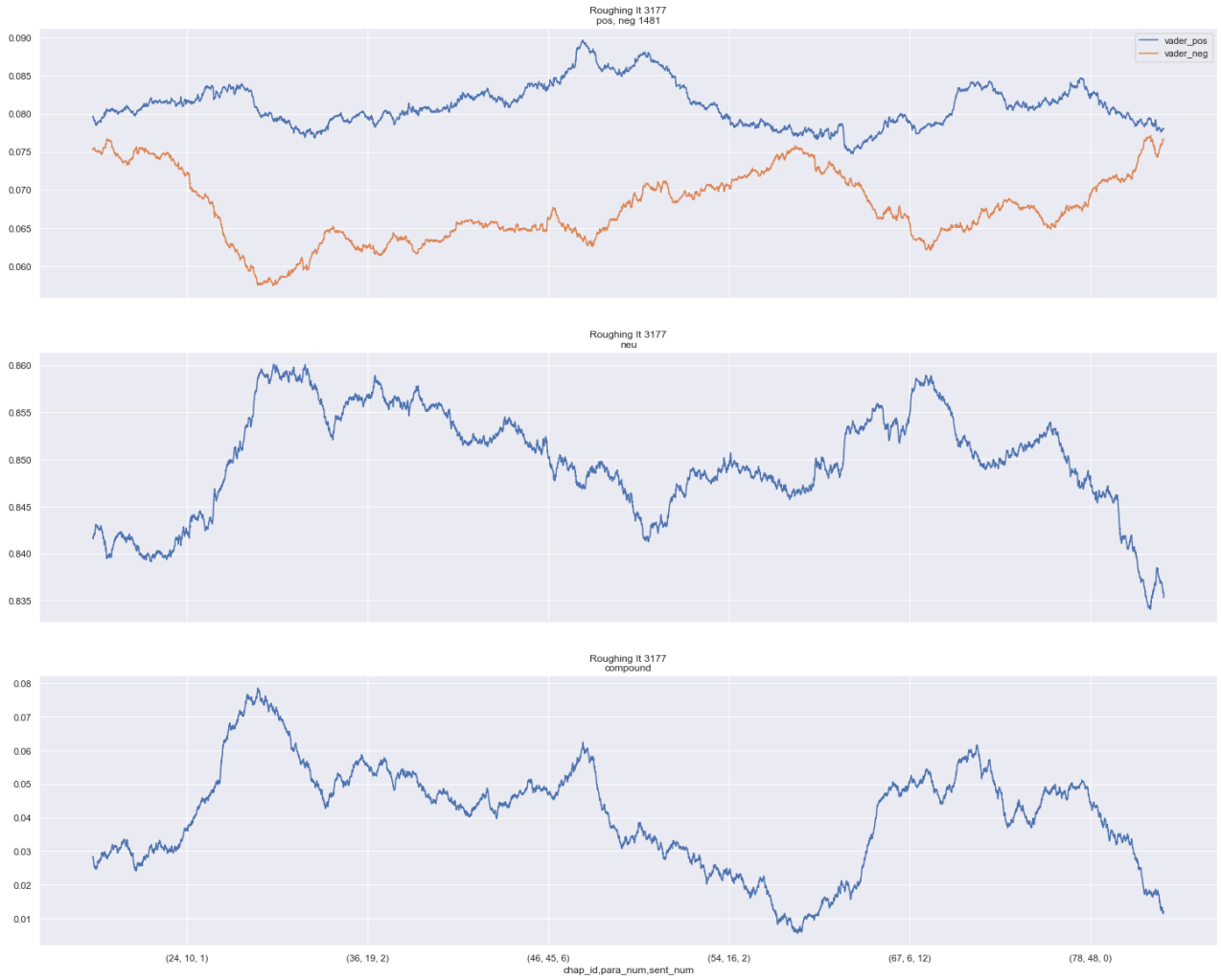
```
vader_plot(bleak)
```



In [55]:
```
vader_plot(sawyer)
```



In [56]:
```
vader_plot(rough)
```

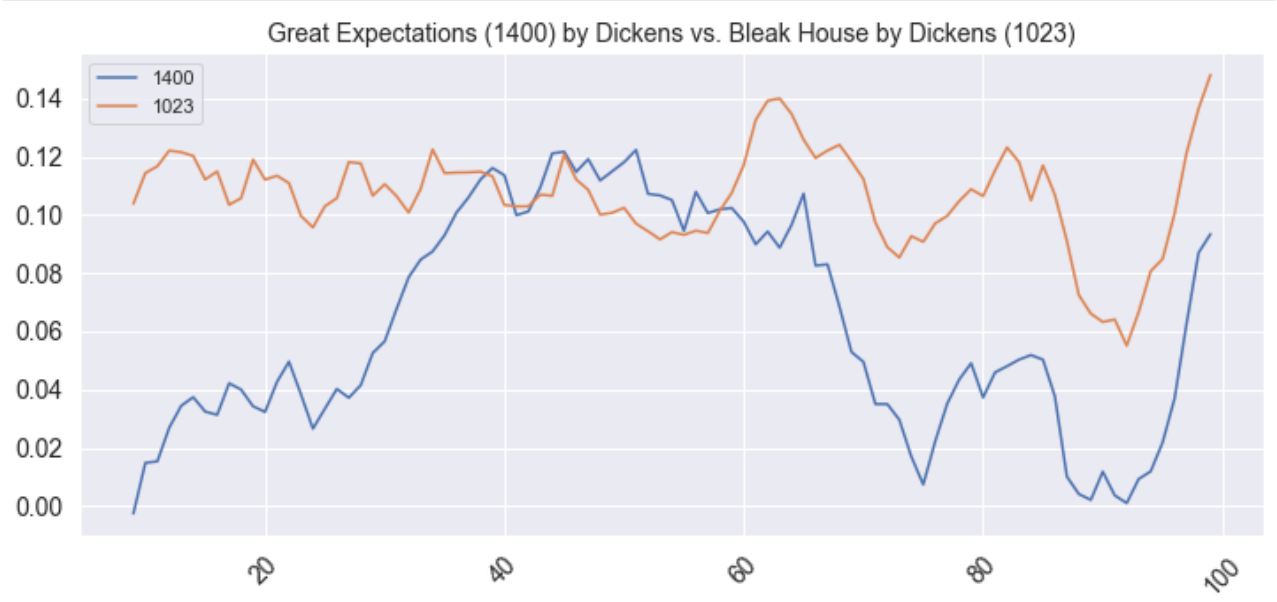


Compare Novels

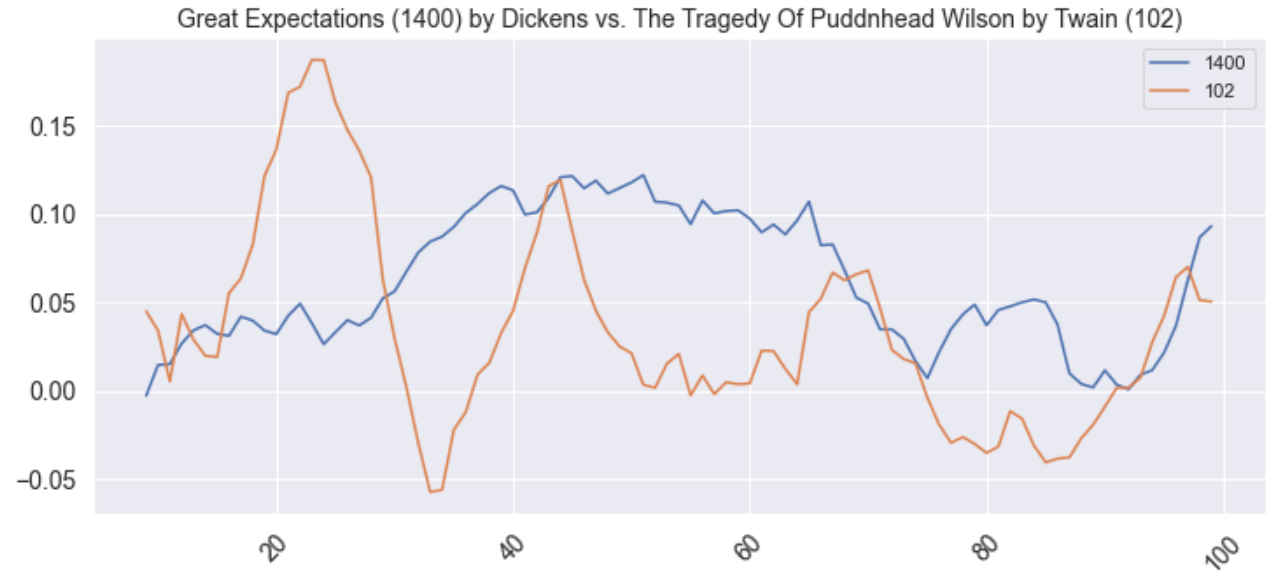
In [57]:
```
FIG = dict(figsize=(12, 5), legend=True, fontsize=14, rot=45)
```

In [58]:
```
def compare_novels(novel_a, novel_b, w=10, emo='vader_compound'):
    global SENTENCES, FIG

    A = SENTENCES.loc[novel_a].reset_index(drop=True).reset_index().rename(colum
    A['cut'] = pd.cut(A.seq, 100)
    A1 = A.groupby('cut')[emo].mean().reset_index(drop=True)

    B = SENTENCES.loc[novel_b].reset_index(drop=True).reset_index().rename(colum
    B['cut'] = pd.cut(B.seq, 100)
    B1 = B.groupby('cut')[emo].mean().reset_index(drop=True)

    C = pd.concat([A1,B1], axis=1)
    C.columns = [novel_a, novel_b]

    plt = C.rolling(w).mean().plot(**FIG)
    plt.set_title(f'{LIB.loc[novel_a, "title"].title()} ({novel_a}) by {LIB.loc[
                  fontsize = 14);
```

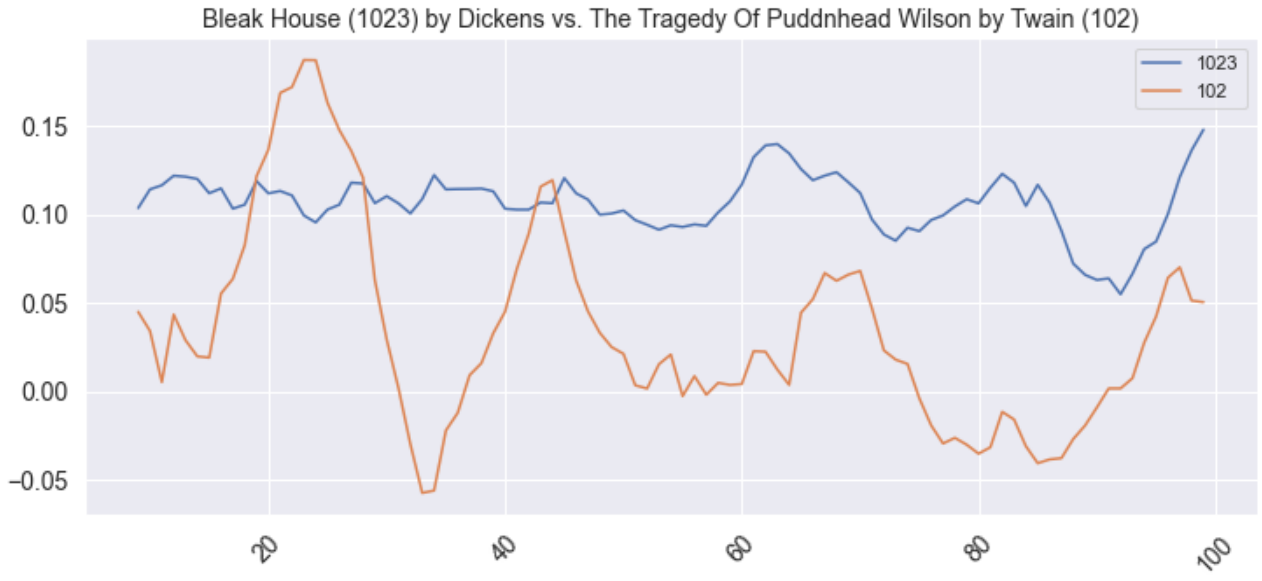
In [59]:
```
compare_novels(great_expectations, bleak)
```



In [60]:
```
compare_novels(great_expectations, puddnhead)
```



In [61]:
```
compare_novels(bleak, puddnhead)
```

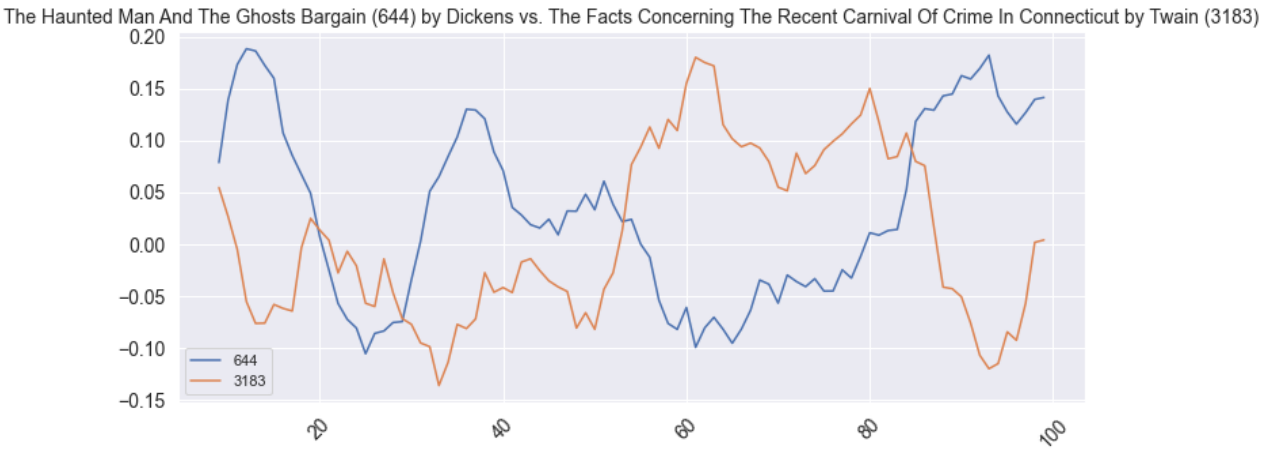


In [63]:

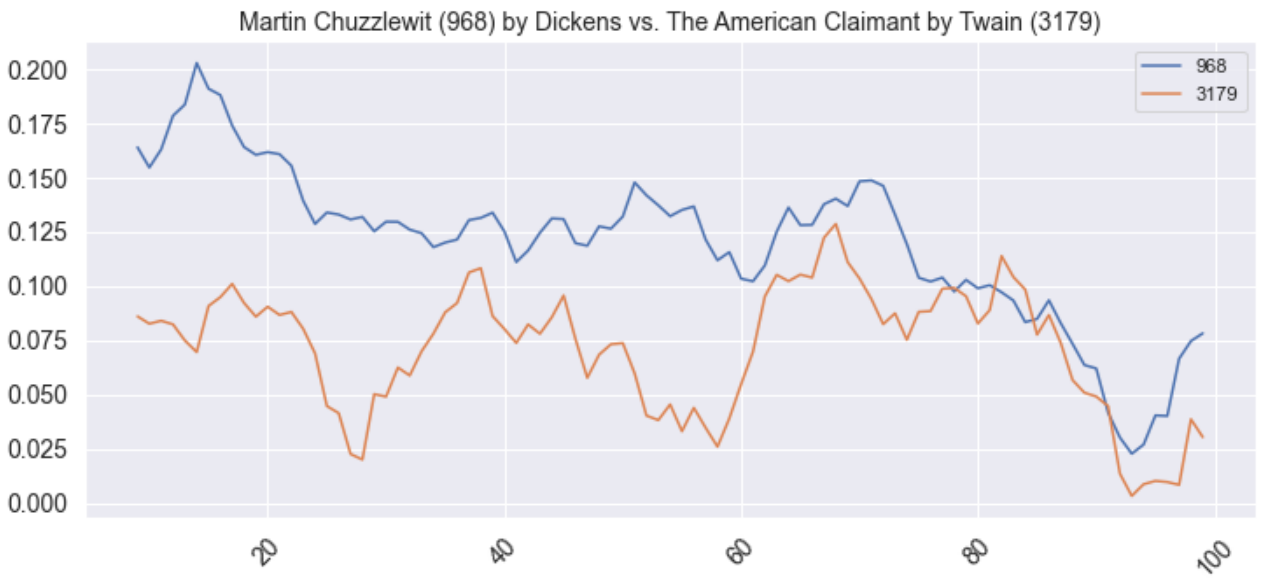
```
# additional dickens novels for comparison
haunted = novels.dickens_the_haunted_man_and_the_ghosts_bargain_644
martin = novels.dickens_martin_chuzzlewit_968
curious = novels.dickens_the_old_curiosity_shop_700
dombey = novels.dickens_dombey_and_sons_821
# travel books
american_notes = novels.dickens_american_notes_675
italy = novels.dickens_pictures_from_italy_650
traveller = novels.dickens_the_uncommerical_traveller_914


# additional twain novels for comparison
carnival = novels.twain_the_facts_concerning_the_recent_carnival_of_crime_in_con
pauper = novels.twain_the_prince_and_the_pauper_1837
# travel books
claimant = novels.twain_the_american_claimant_3179
innocents = novels.twain_the_innocents_abroad_3176
tramp = novels.twain_a_tramp_abroad_119
equator = novels.twain_following_the_equator_2895
miss = novels.twain_life_on_the_mississippi_245
```

In [64]:

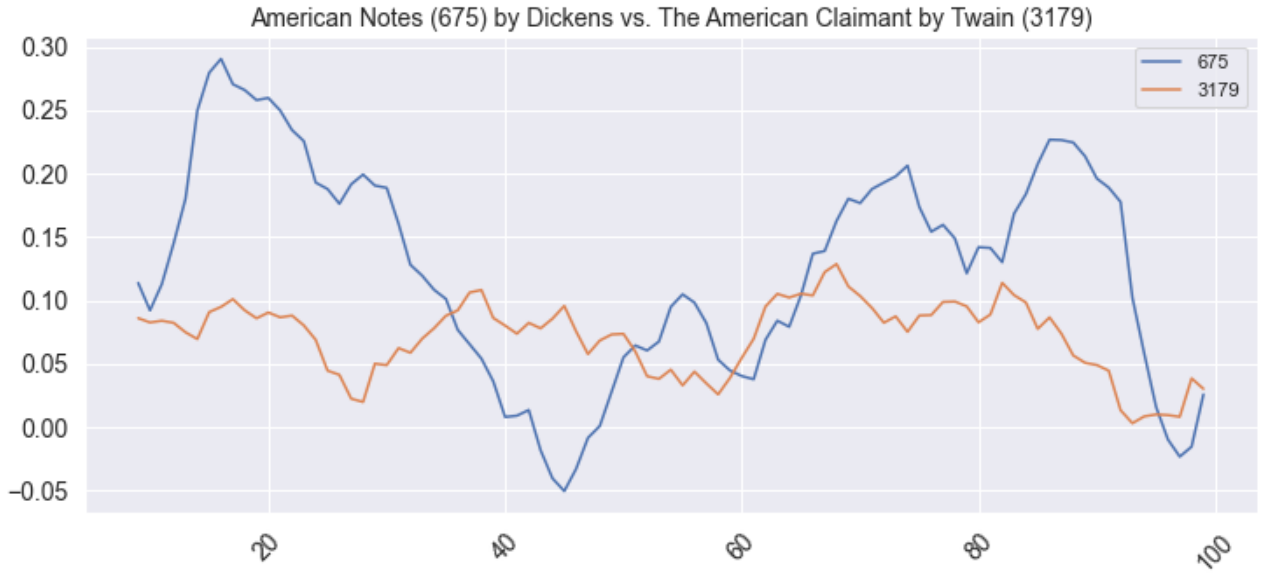
```
compare_novels(haunted, carnival) # eerier stories (??)
```



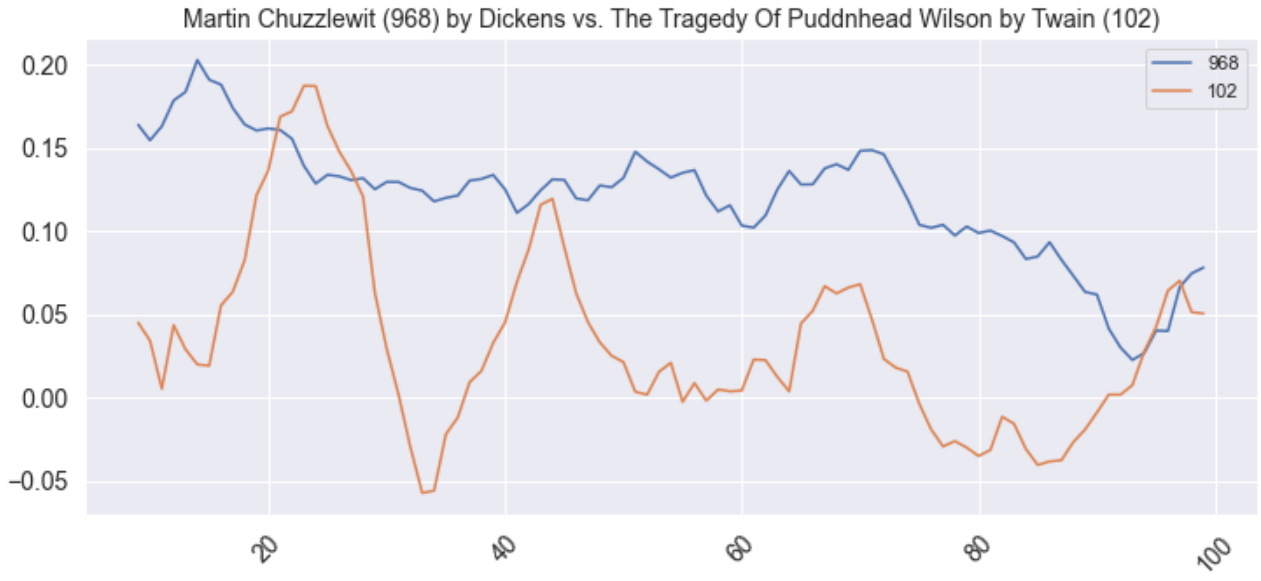
In [65]:
```
compare_novels(martin, claimant) # books about america
```



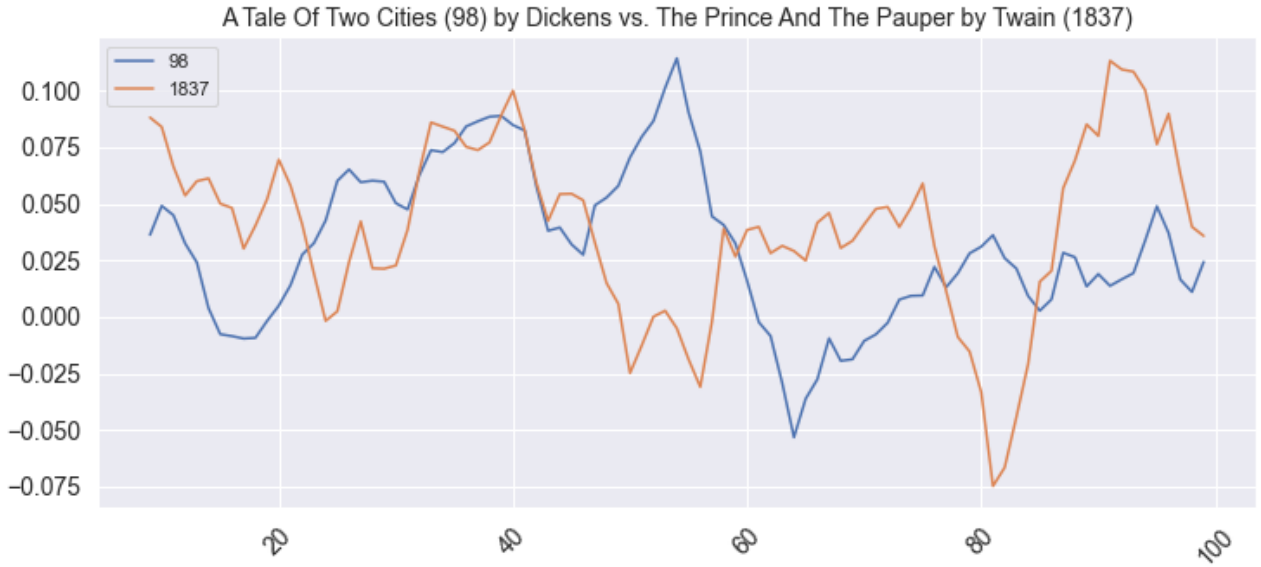
In [66]:
```
compare_novels(american_notes, claimant) # books about america
```



In [90]:
```
compare_novels(martin, puddnhead) # books about america
```



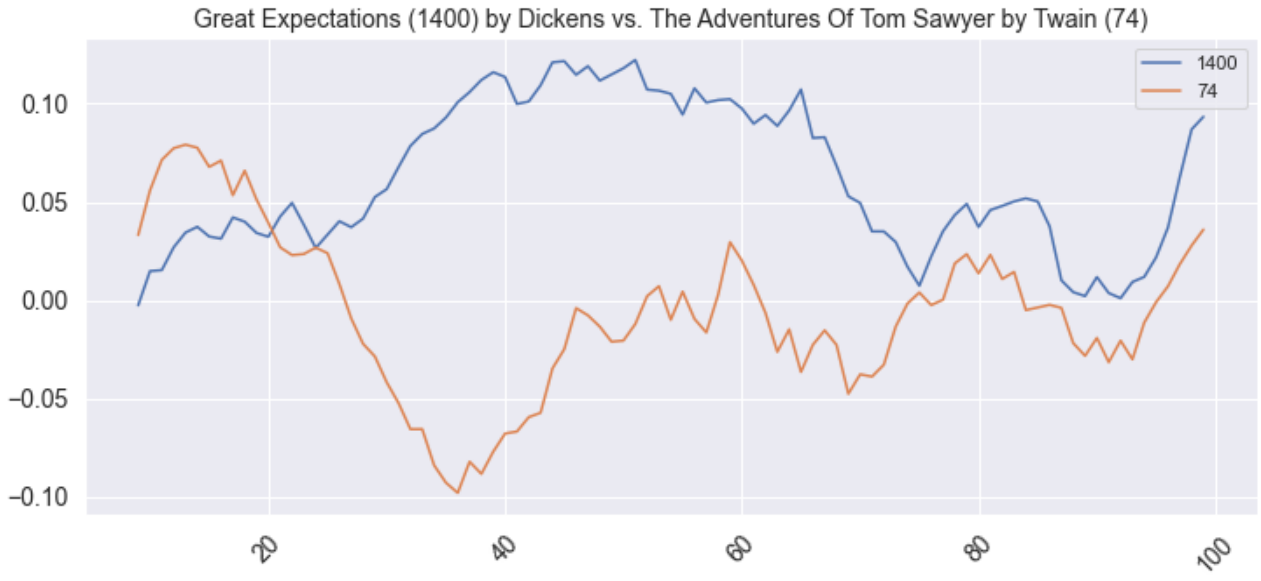
In [67]:
```
compare_novels(two_cities, pauper) # twins
```

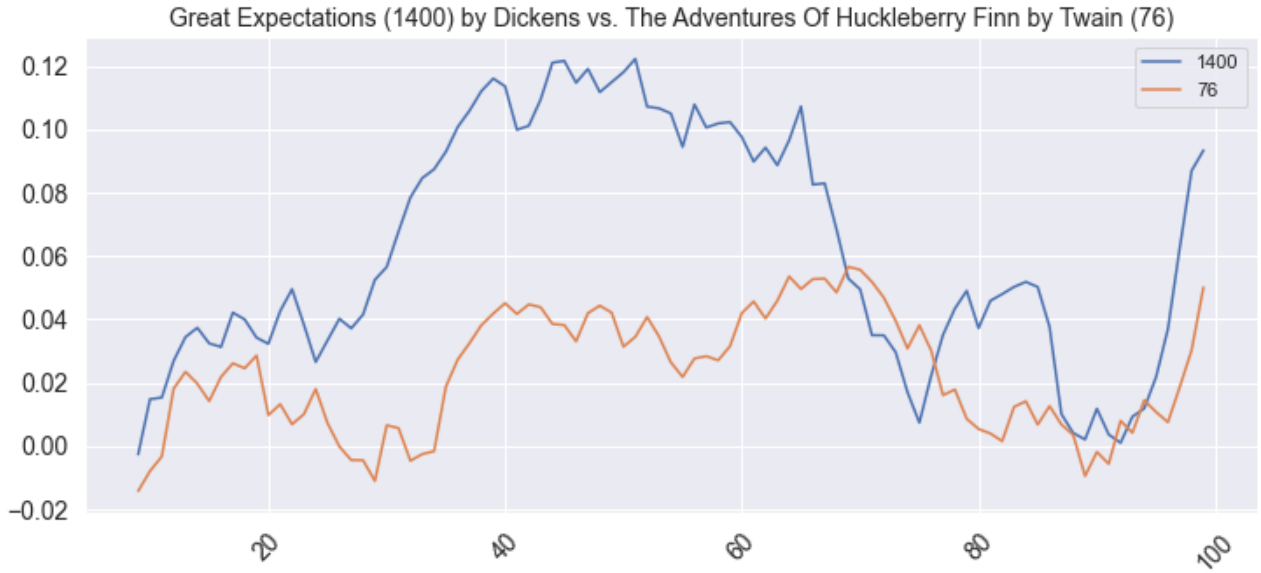


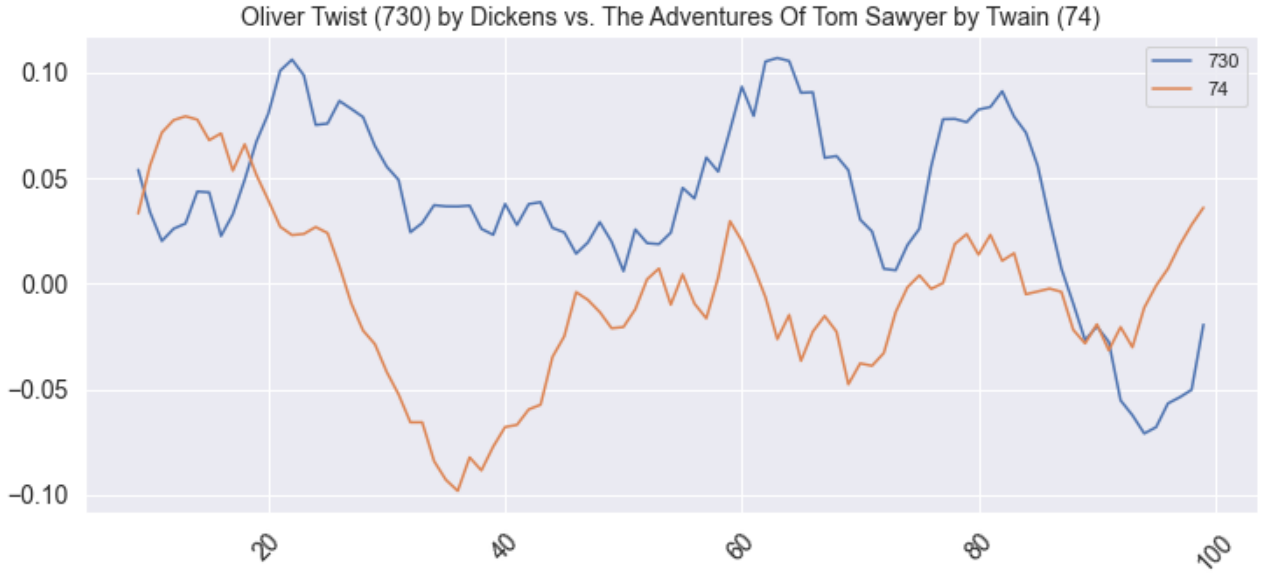
Books with Young Protagonists

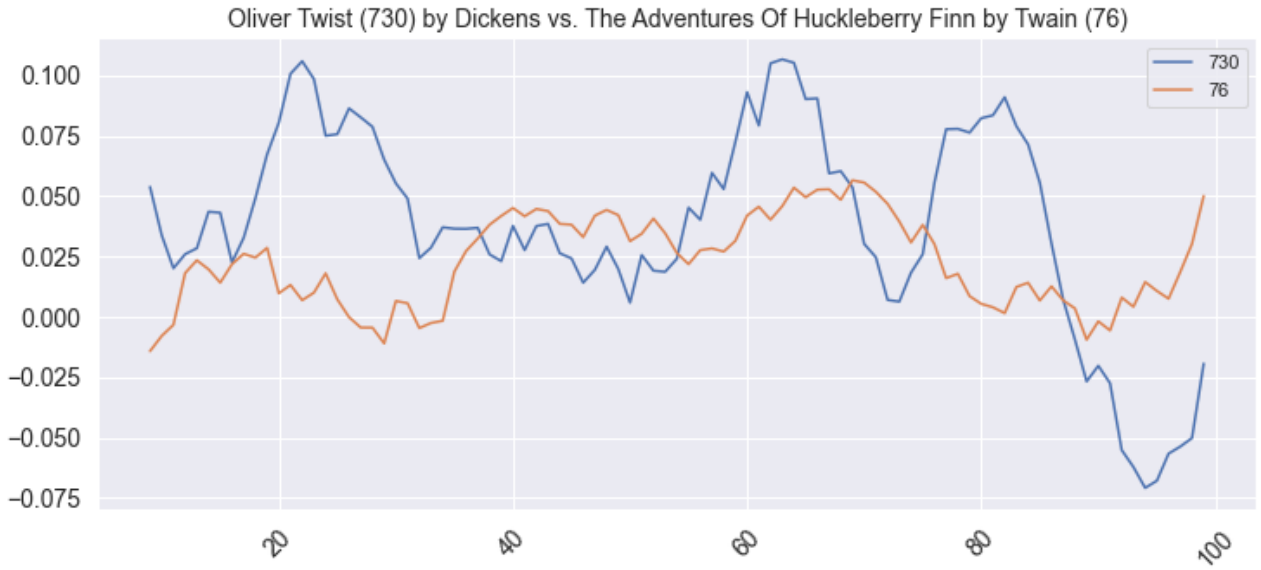
In [68]:
```
# books involving young protagonists

dickens_compare = [great_expectations, twist, curious, copperfield, dombey]

twain_compare = [sawyer, huck]

novel_pairs = [i for i in itertools.combinations(dickens_compare + twain_compare
```

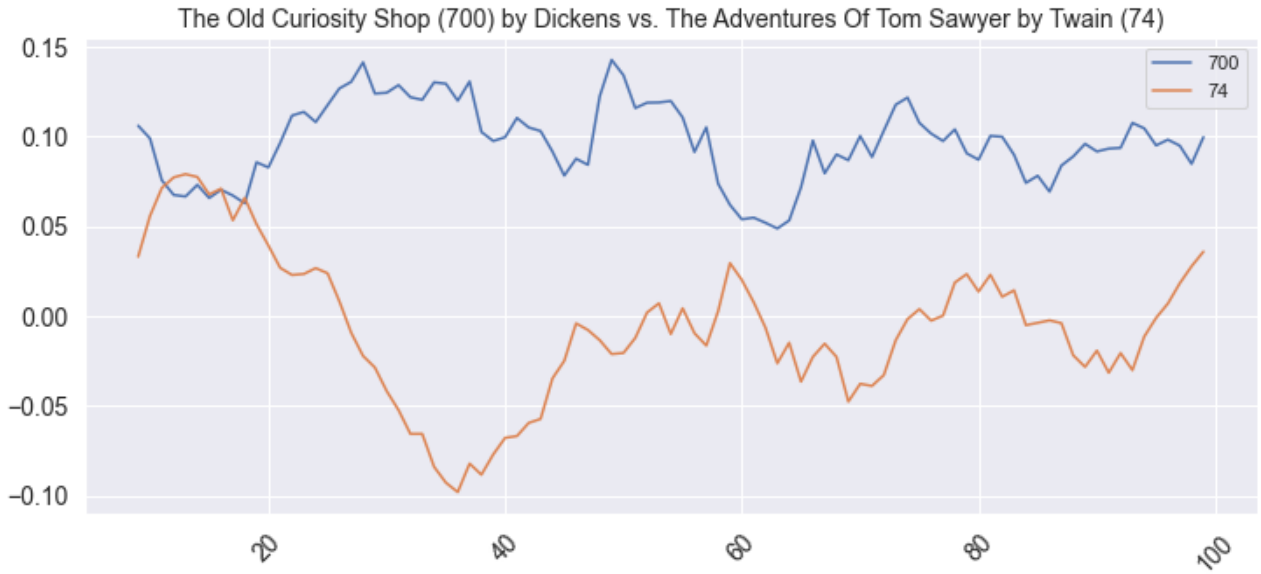
In [69]:
```
for pair in novel_pairs:
    compare_novels(pair[0], pair[1])
```

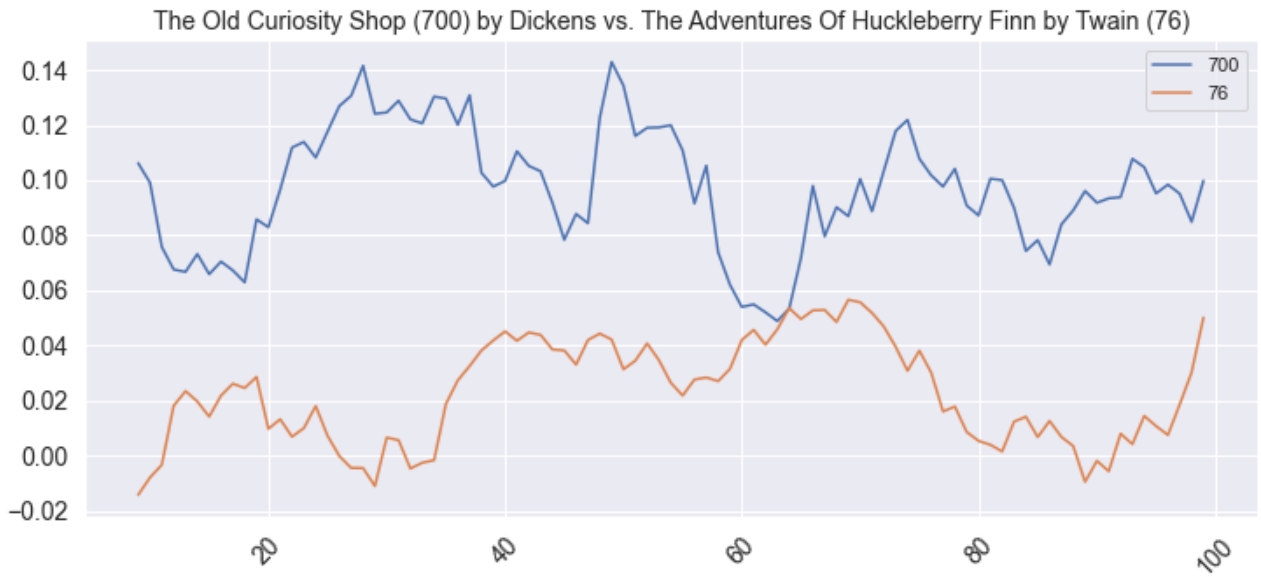

Great Expectations (1400) by Dickens vs. The Adventures Of Tom Sawyer by Twain (74)
1400
74
0.10
0.05
0.00
−0.05
−0.10
20
40
60
80
100

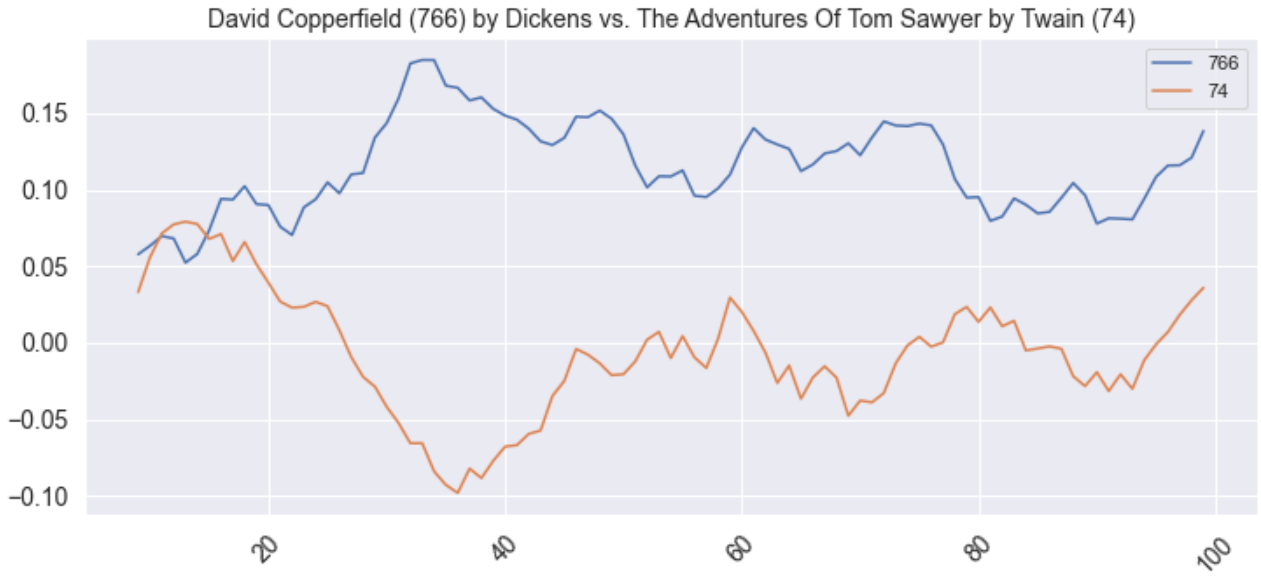

Great Expectations (1400) by Dickens vs. The Adventures Of Huckleberry Finn by Twain (76)
1400
76
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−0.02
20
40
60
80
100

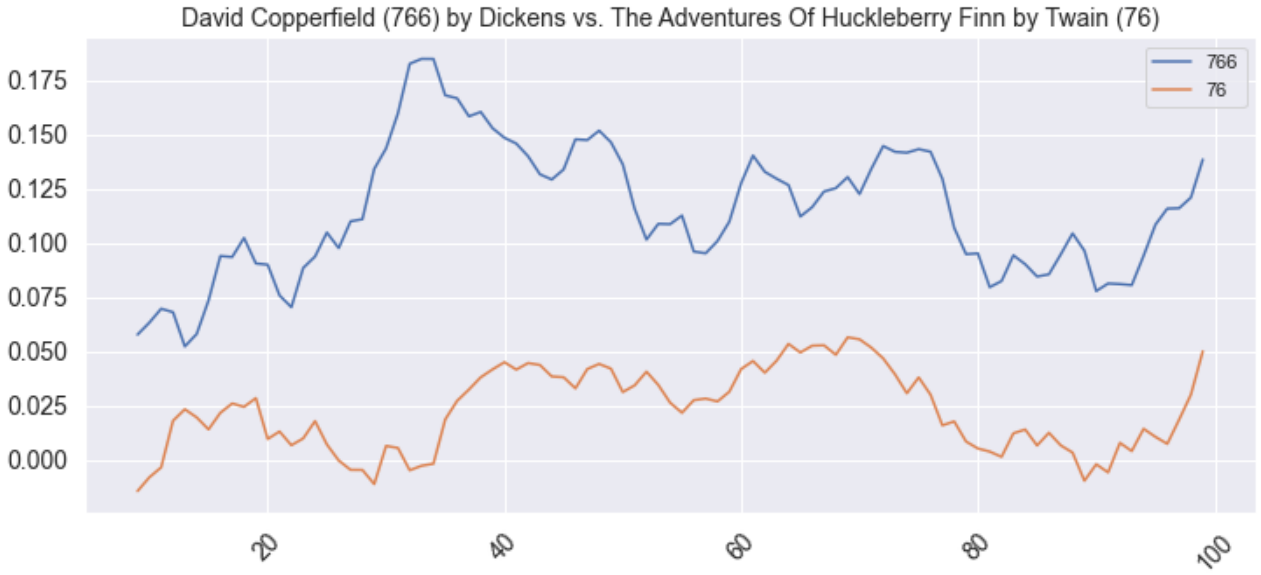

Oliver Twist (730) by Dickens vs. The Adventures Of Tom Sawyer by Twain (74)
730
74
0.10
0.05
0.00
−0.05
−0.10
20
40
60
80
100

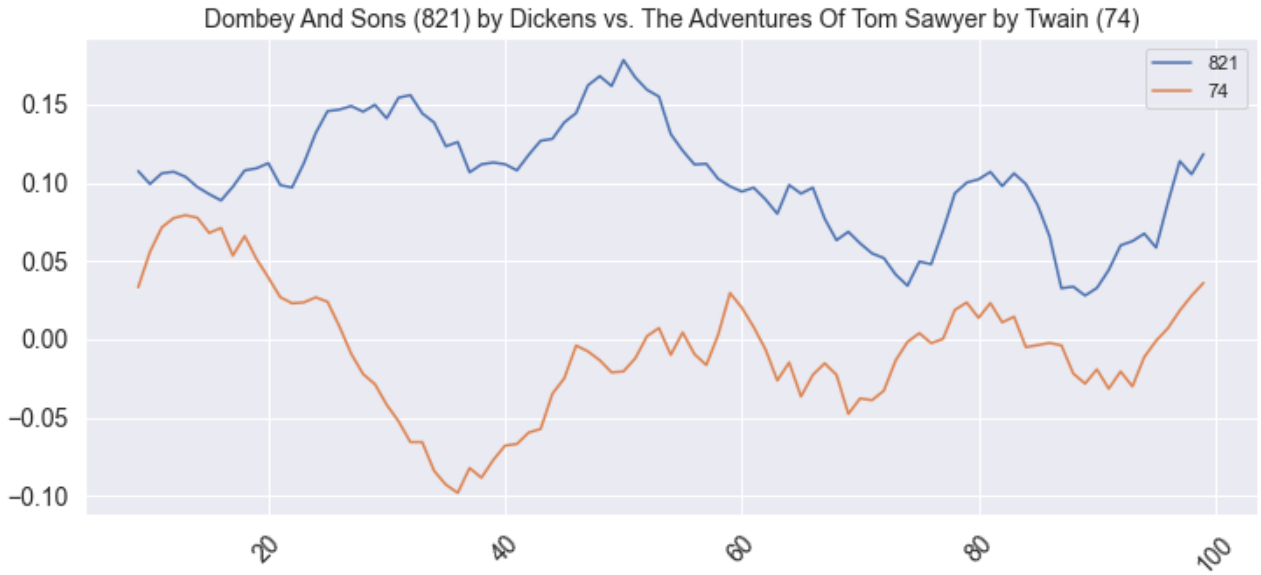

Oliver Twist (730) by Dickens vs. The Adventures Of Huckleberry Finn by Twain (76)
730
76
0.100
0.075
0.050
0.025
0.000
−0.025
−0.050
−0.075
20
40
60
80
100

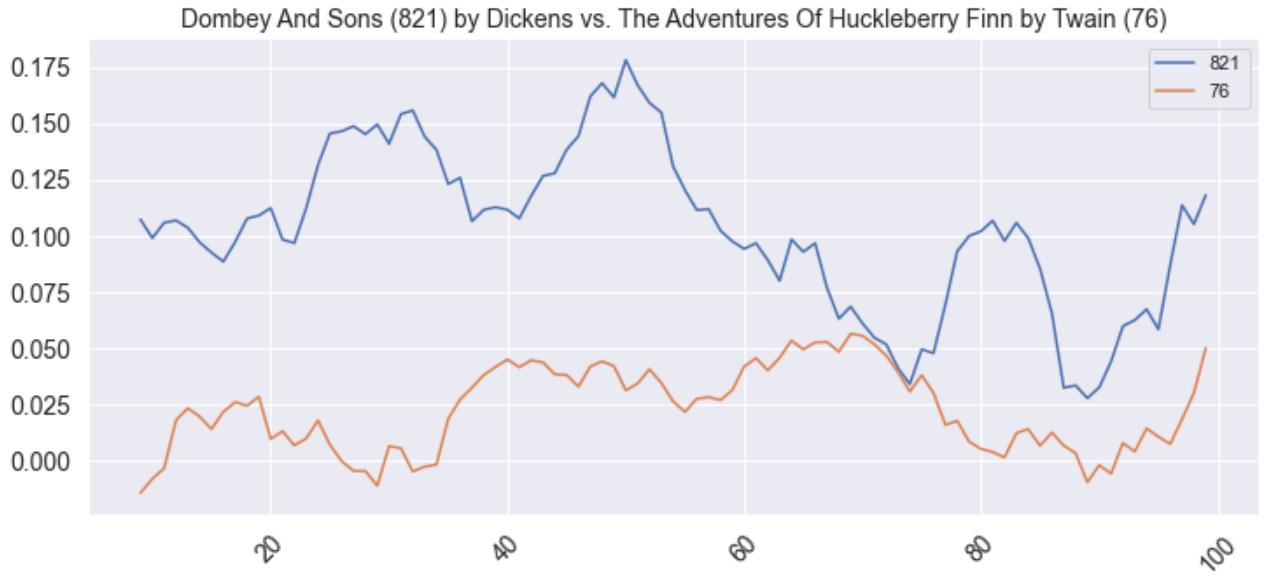

The Old Curiosity Shop (700) by Dickens vs. The Adventures Of Tom Sawyer by Twain (74)
700
74
0.15
0.10
0.05
0.00
−0.05
−0.10
20
40
60
80
100


The Old Curiosity Shop (700) by Dickens vs. The Adventures Of Huckleberry Finn by Twain (76)
700
76
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−0.02
20
40
60
80
100


David Copperfield (766) by Dickens vs. The Adventures Of Tom Sawyer by Twain (74)
766
74
0.15
0.10
0.05
0.00
−0.05
−0.10
20
40
60
80
100


David Copperfield (766) by Dickens vs. The Adventures Of Huckleberry Finn by Twain (76)
766
76
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
20
40
60
80
100

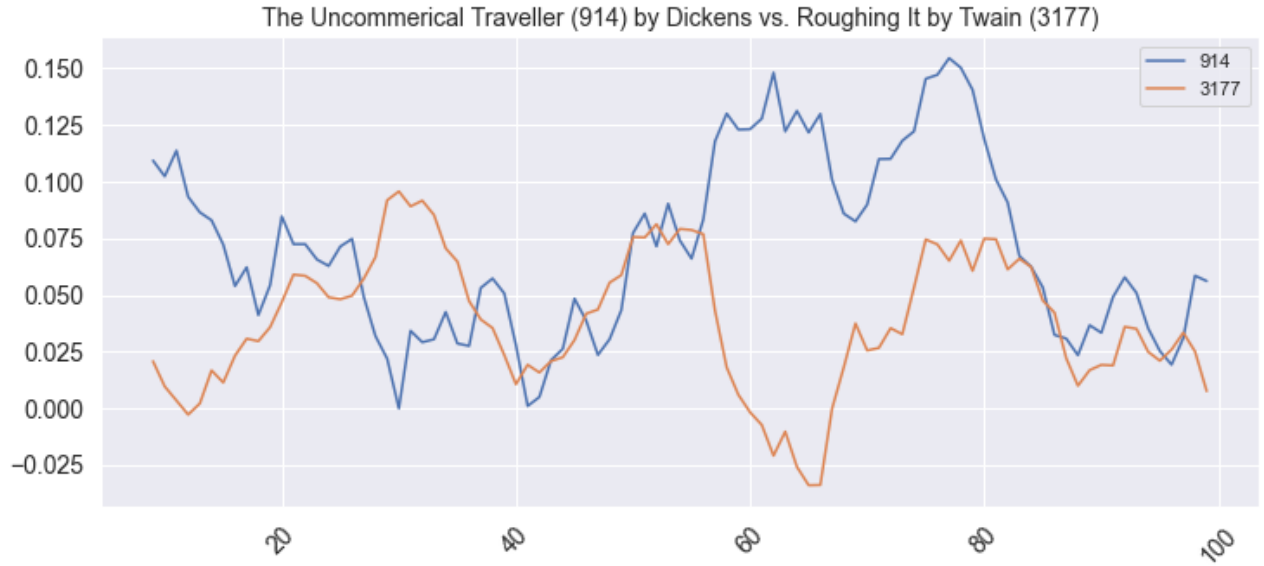

Dombey And Sons (821) by Dickens vs. The Adventures Of Tom Sawyer by Twain (74)
821
74
0.15
0.10
0.05
0.00
−0.05
−0.10
20
40
60
80
100

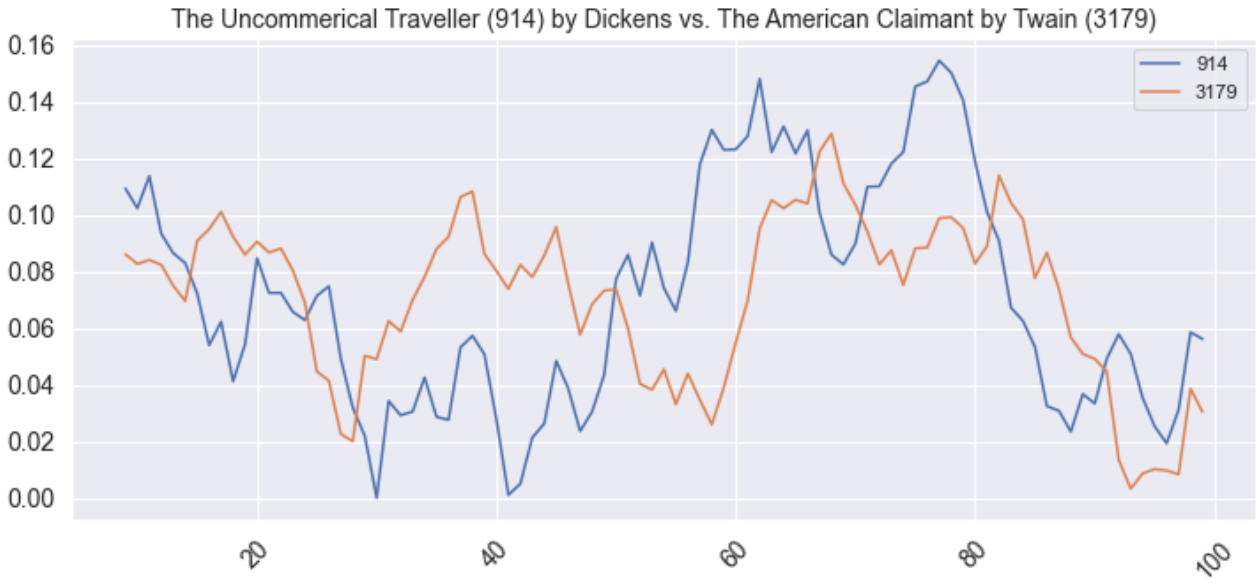


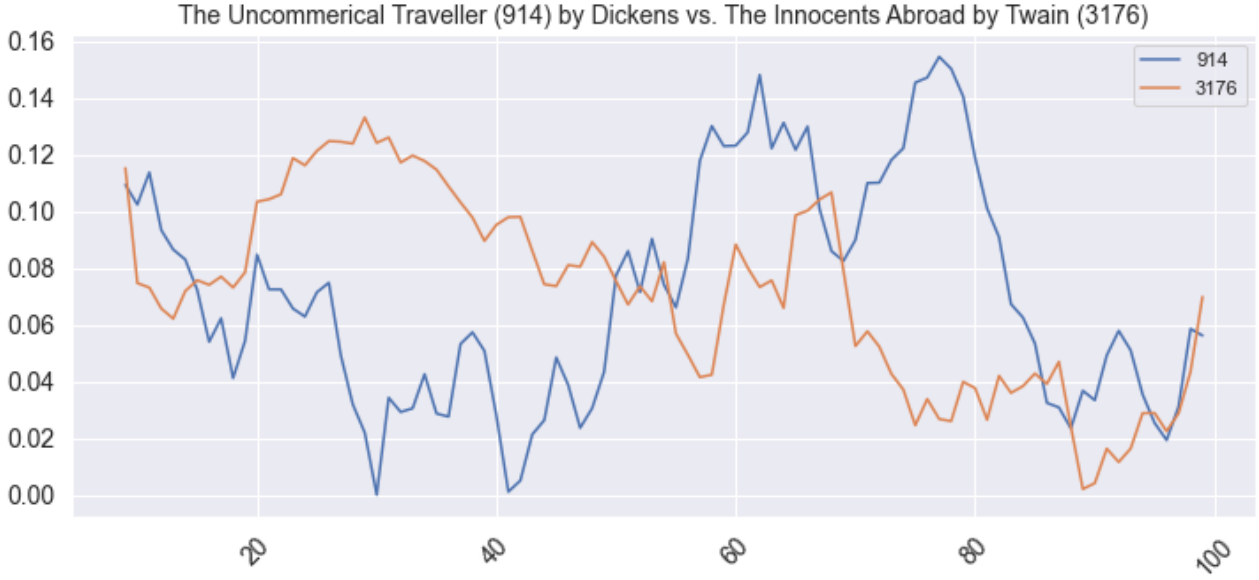
Travel Stories

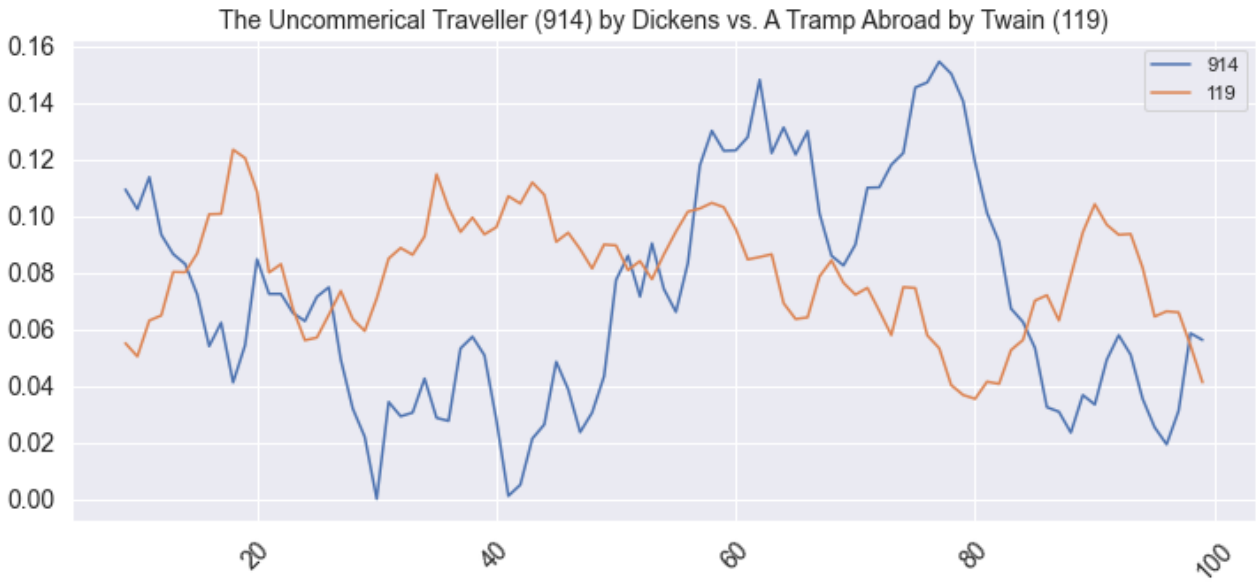
In [70]:
```
dickens_travel = [traveller, american_notes, italy]

twain_travel = [rough, claimant, innocents, tramp, equator, miss]

travel_pairs = [i for i in itertools.combinations(dickens_travel + twain_travel,
```

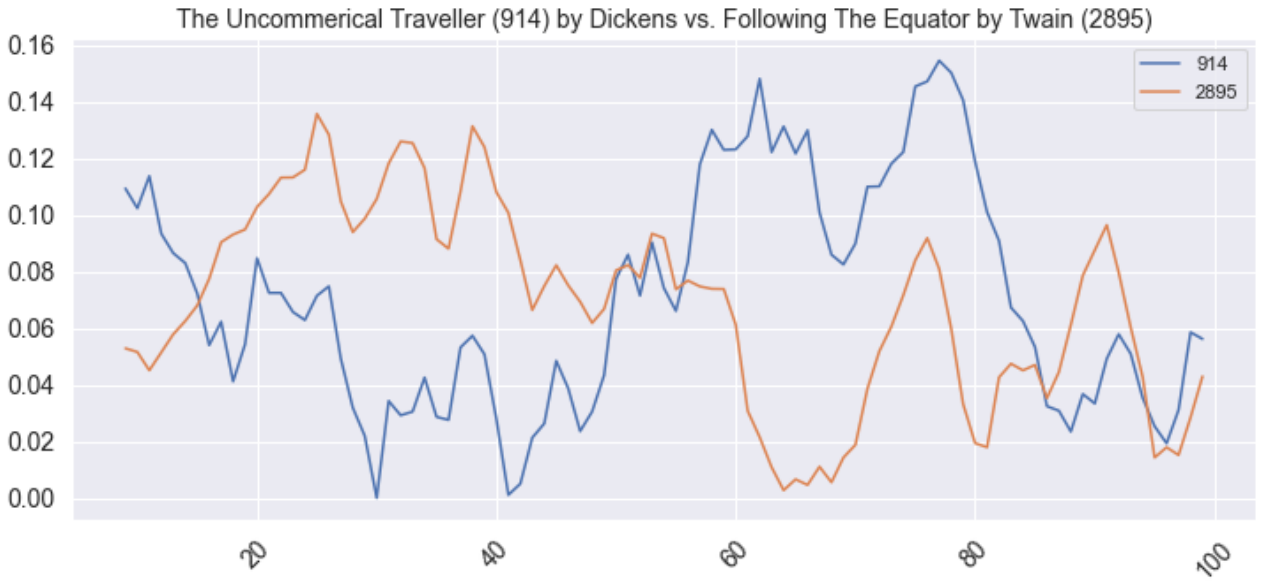
In [71]:
```
for pair in travel_pairs:
    compare_novels(pair[0], pair[1])
```

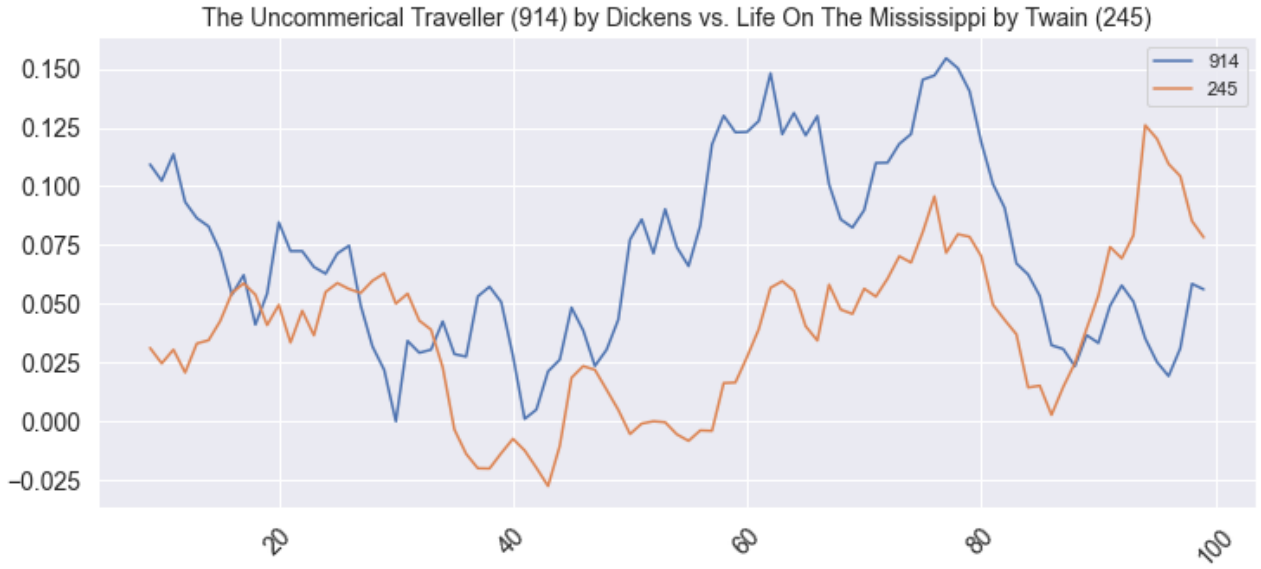


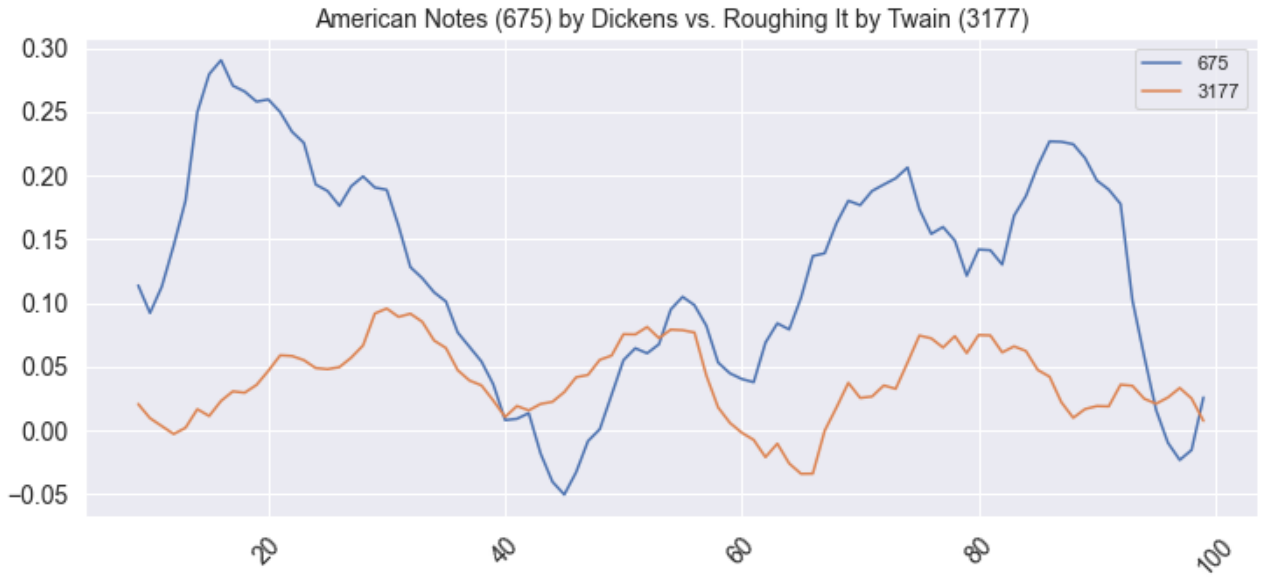

The Uncommerical Traveller (914) by Dickens vs. The American Claimant by Twain (3179)
914
3179
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
20
40
60
80
100

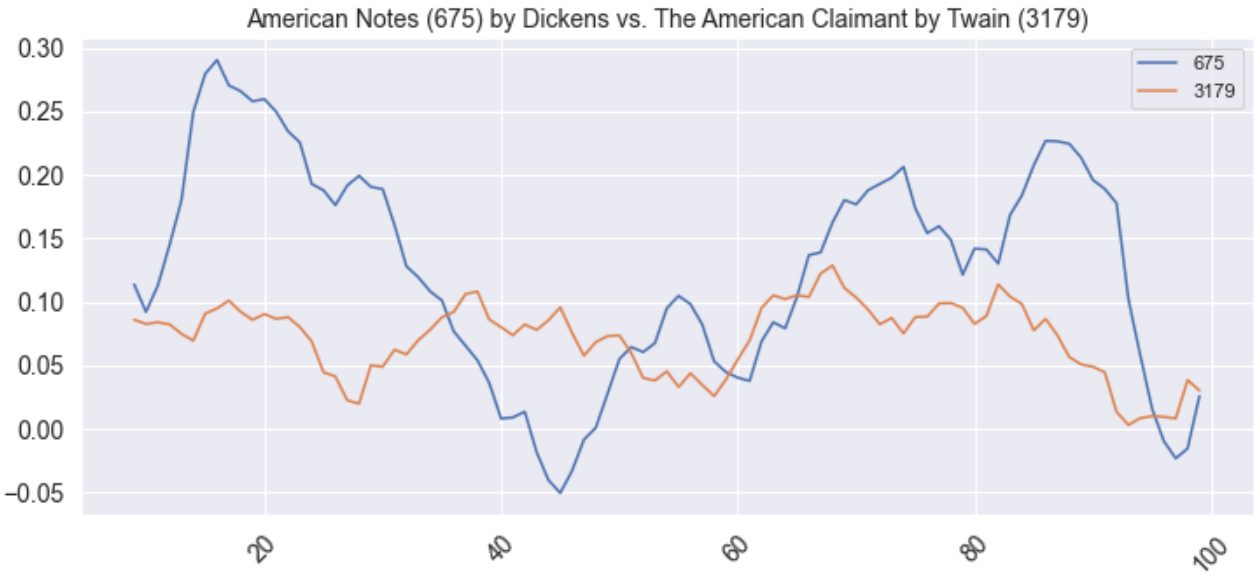

The Uncommerical Traveller (914) by Dickens vs. The Innocents Abroad by Twain (3176)
914
3176
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
20
40
60
80
100

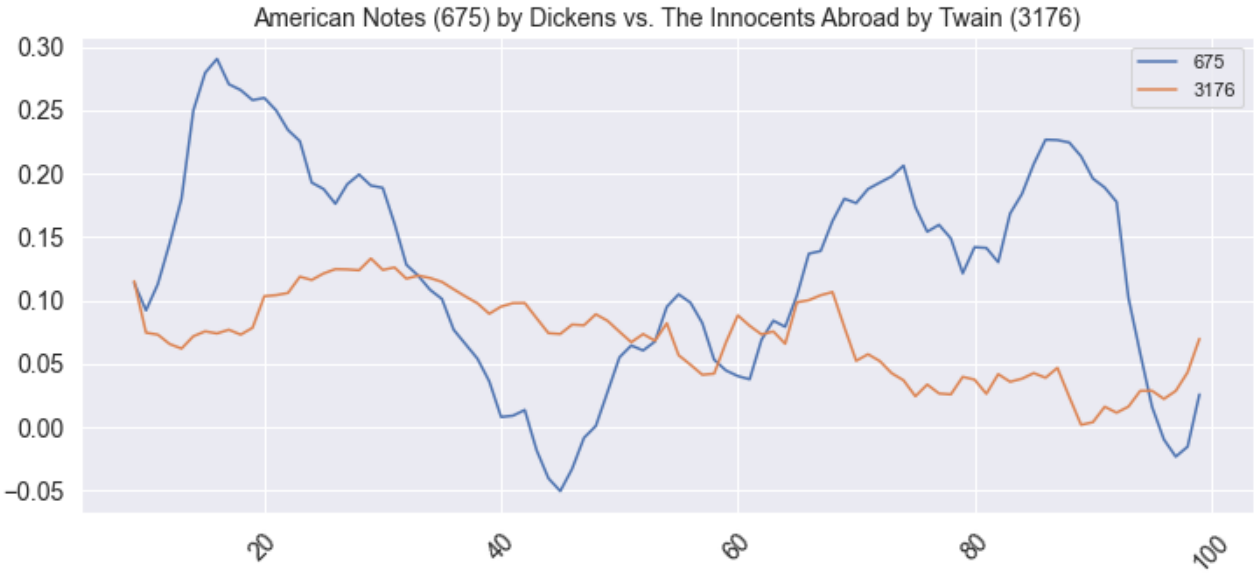

The Uncommerical Traveller (914) by Dickens vs. A Tramp Abroad by Twain (119)
914
119
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
20
40
60
80
100

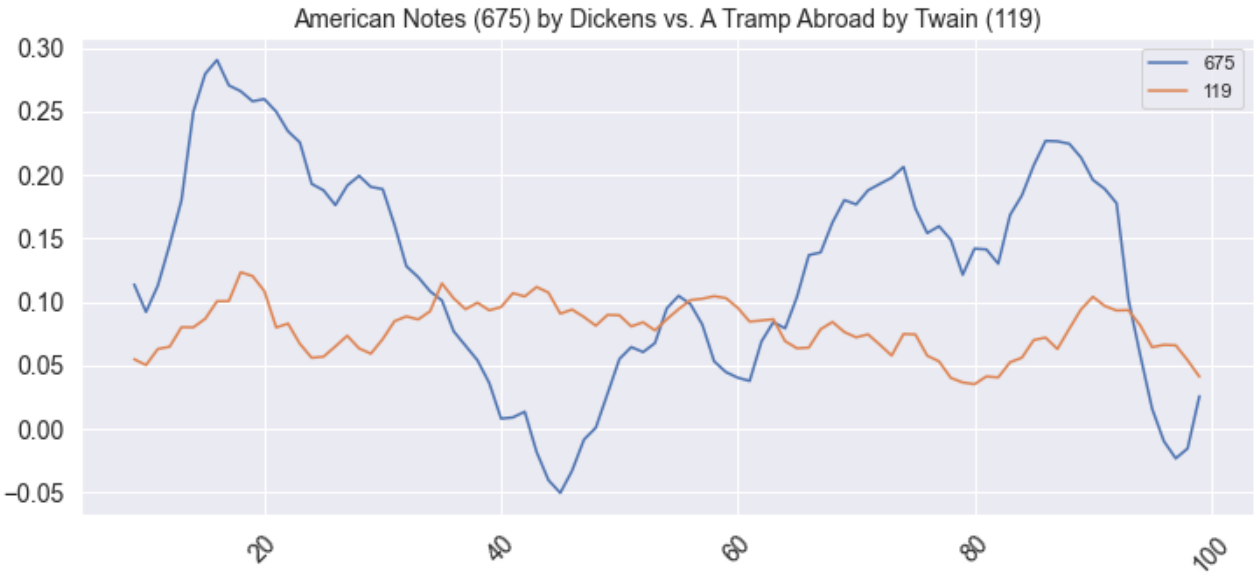

The Uncommerical Traveller (914) by Dickens vs. Following The Equator by Twain (2895)
914
2895
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
20
40
60
80
100

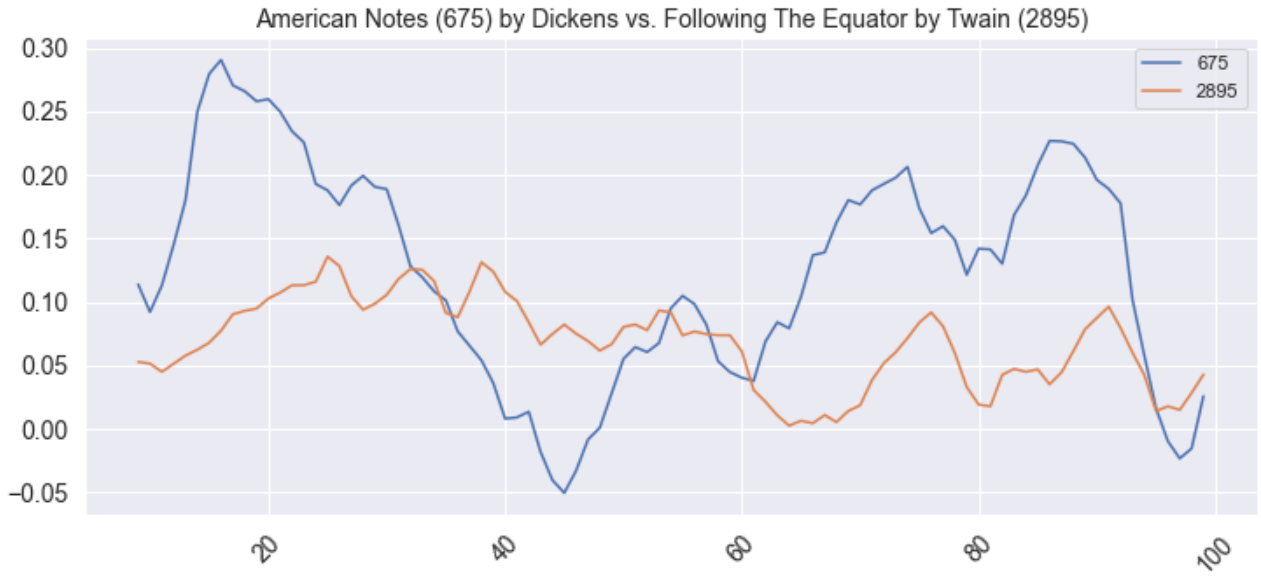

The Uncommerical Traveller (914) by Dickens vs. Life On The Mississippi by Twain (245)
914
245
0.150
0.125
0.100
0.075
0.050
0.025
0.000
−0.025
20
40
60
80
100

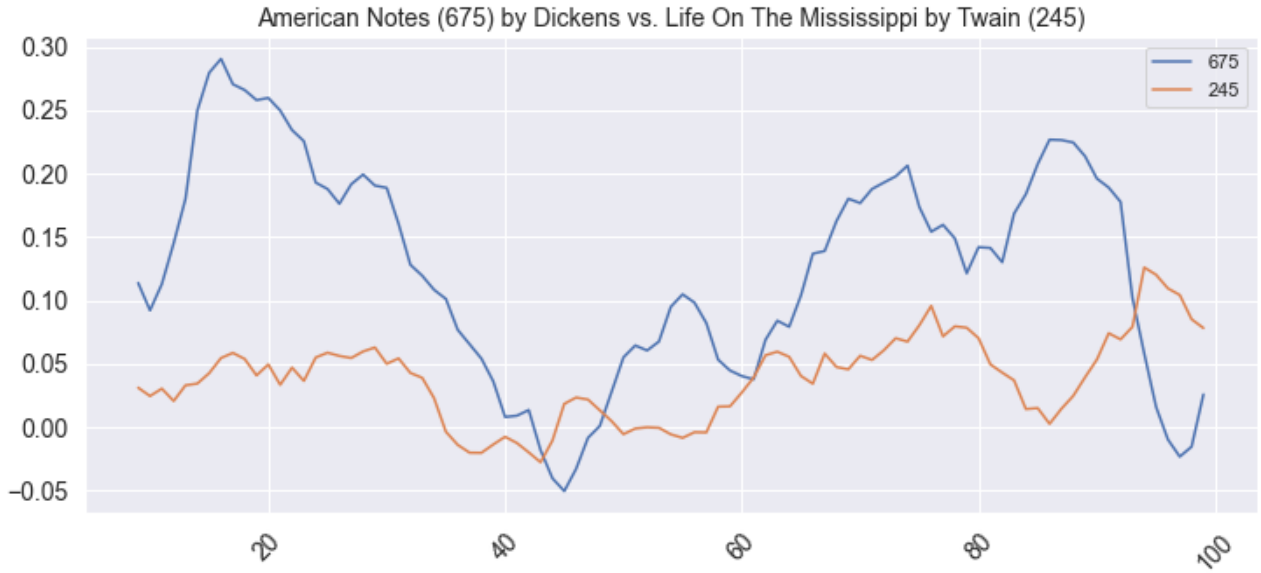

American Notes (675) by Dickens vs. Roughing It by Twain (3177)
675
3177
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
20
40
60
80
100

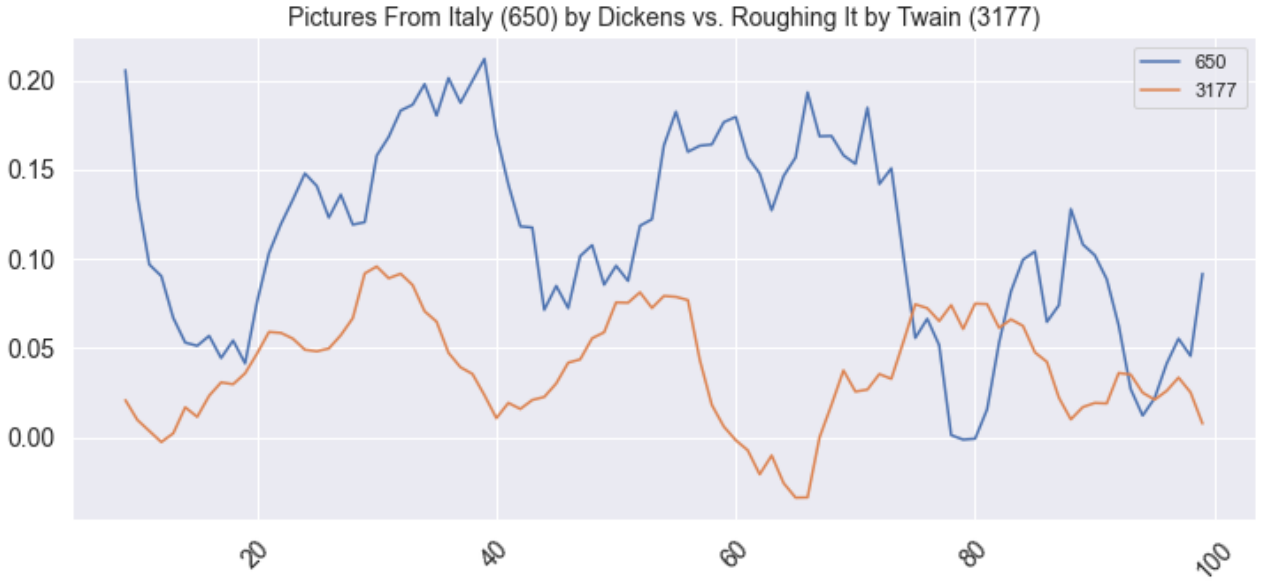

American Notes (675) by Dickens vs. The American Claimant by Twain (3179)
675
3179
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
20
40
60
80
100

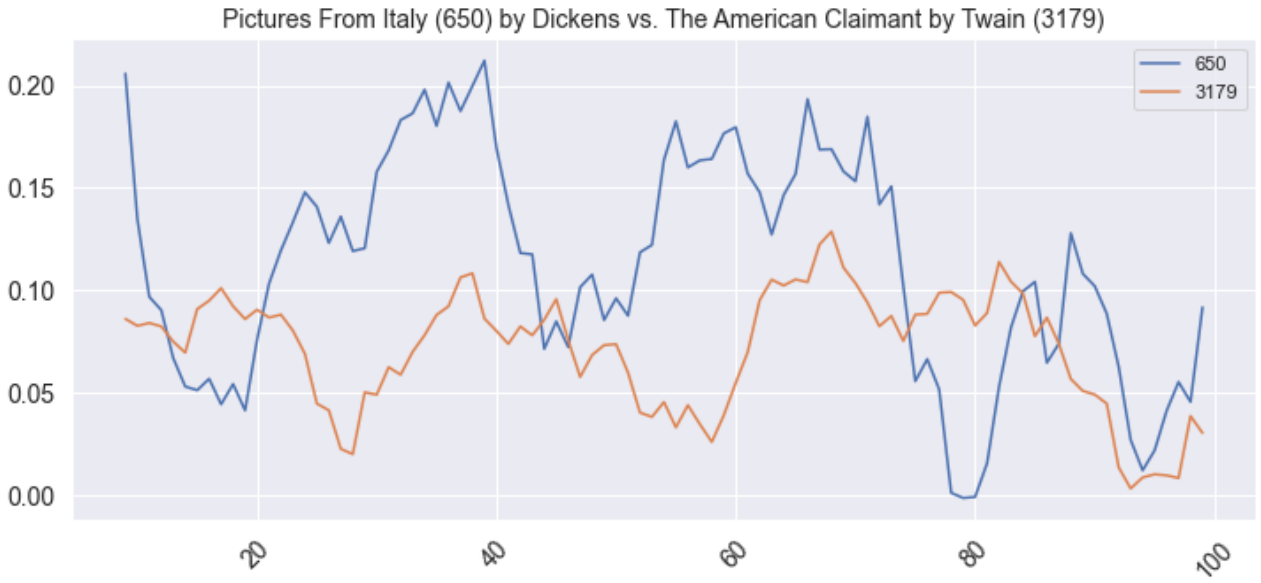

American Notes (675) by Dickens vs. The Innocents Abroad by Twain (3176)
675
3176
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
20
40
60
80
100

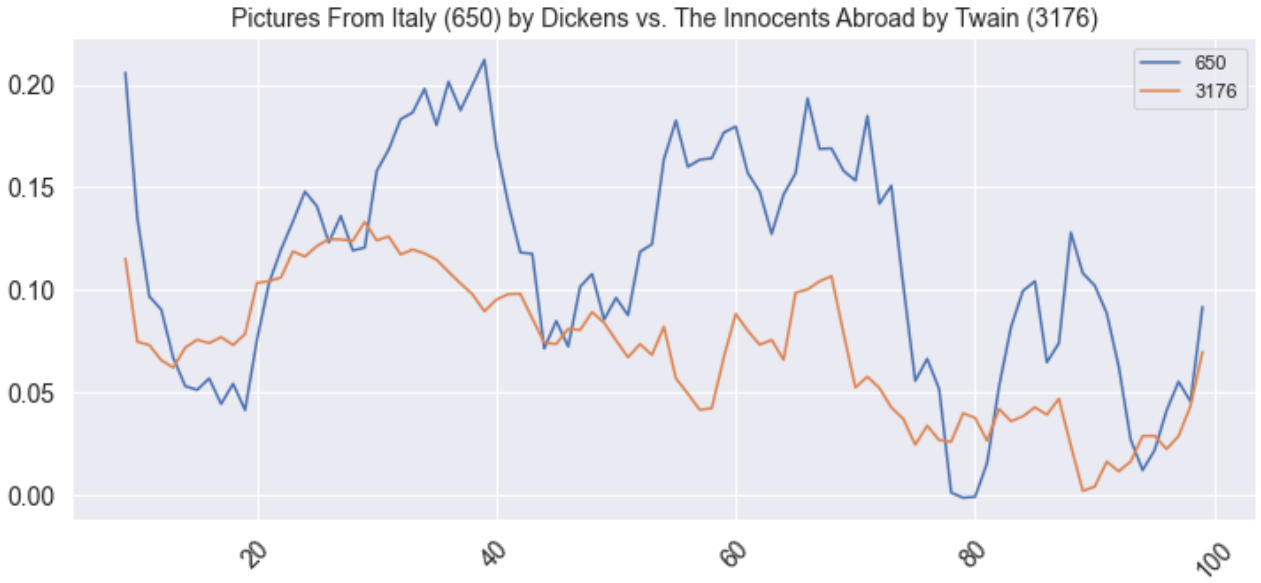

American Notes (675) by Dickens vs. A Tramp Abroad by Twain (119)
675
119
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
20
40
60
80
100

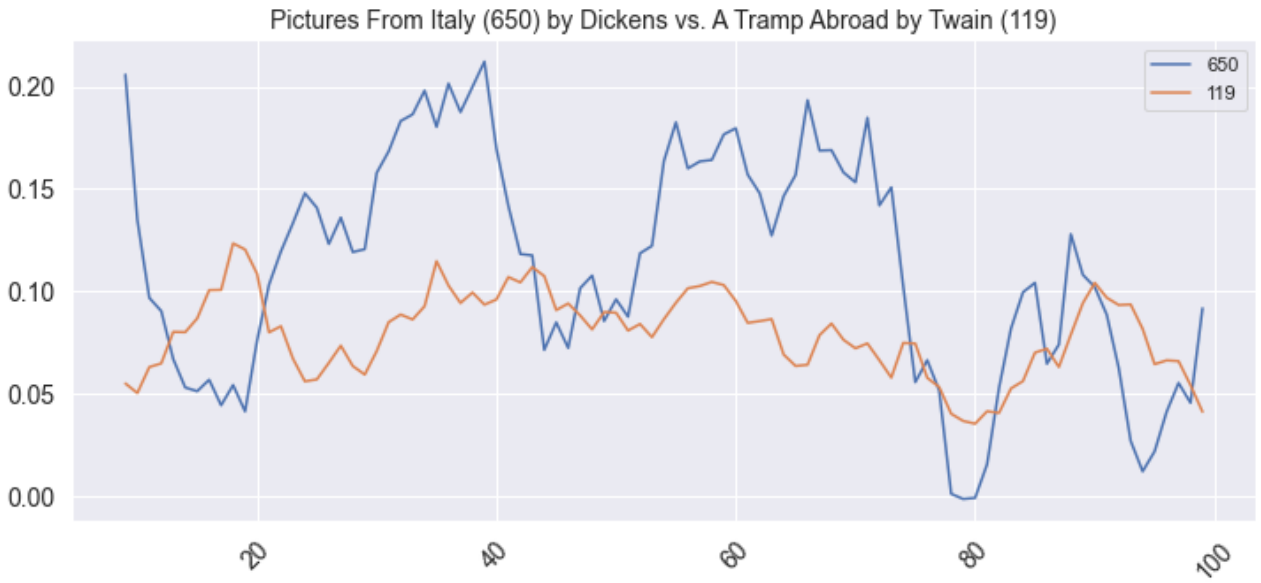

American Notes (675) by Dickens vs. Following The Equator by Twain (2895)
675
2895
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
20
40
60
80
100

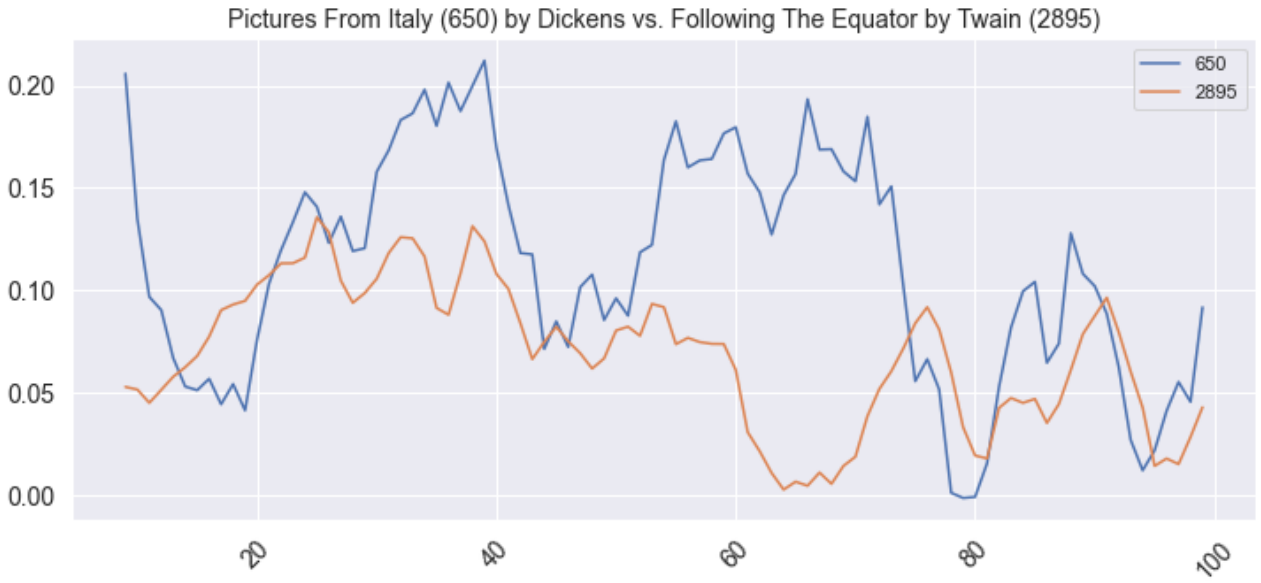

American Notes (675) by Dickens vs. Life On The Mississippi by Twain (245)
675
245
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
20
40
60
80
100

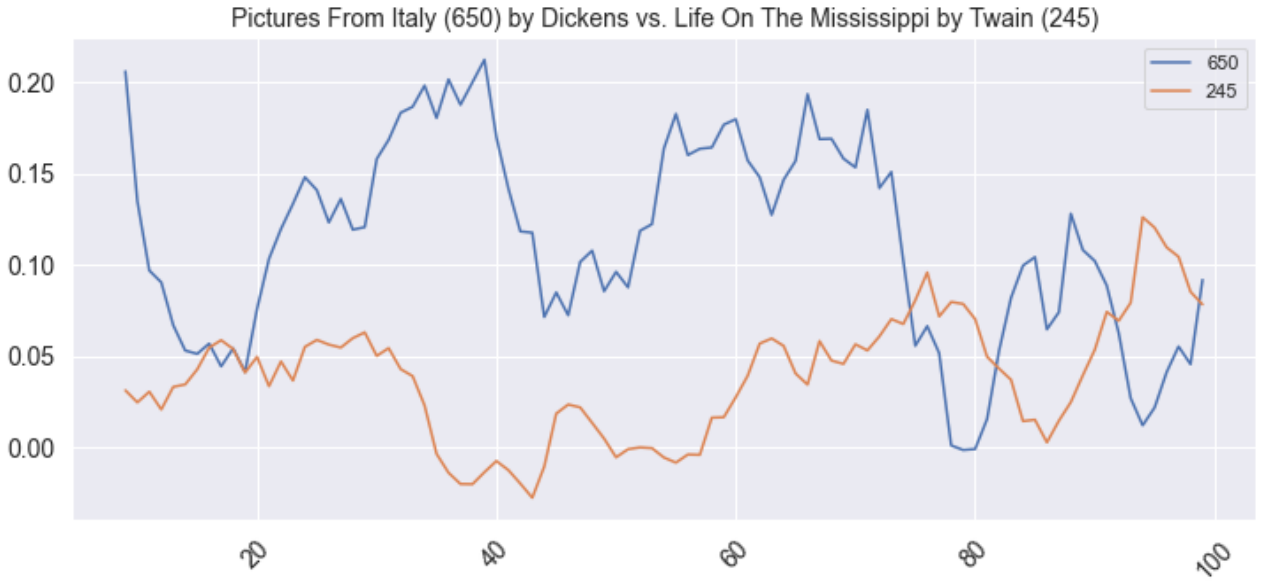

Pictures From Italy (650) by Dickens vs. Roughing It by Twain (3177)
650
3177
0.20
0.15
0.10
0.05
0.00
20
40
60
80
100


Pictures From Italy (650) by Dickens vs. The American Claimant by Twain (3179)
650
3179
0.20
0.15
0.10
0.05
0.00
20
40
60
80
100

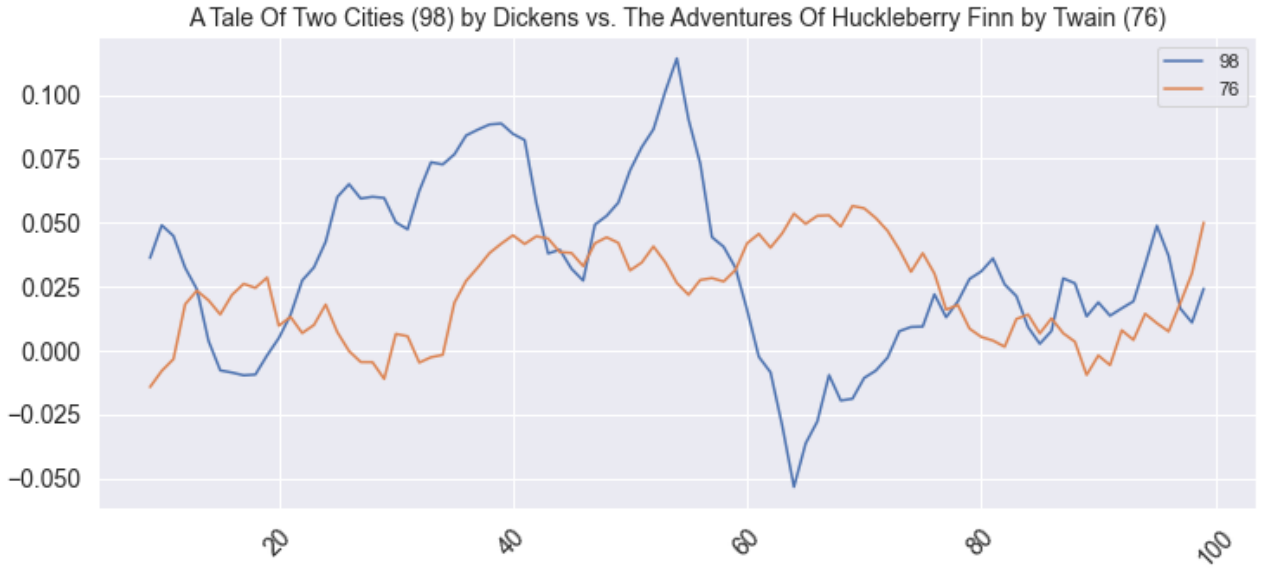

Pictures From Italy (650) by Dickens vs. The Innocents Abroad by Twain (3176)
650
3176
0.20
0.15
0.10
0.05
0.00
20
40
60
80
100


Pictures From Italy (650) by Dickens vs. A Tramp Abroad by Twain (119)
650
119
0.20
0.15
0.10
0.05
0.00
20
40
60
80
100

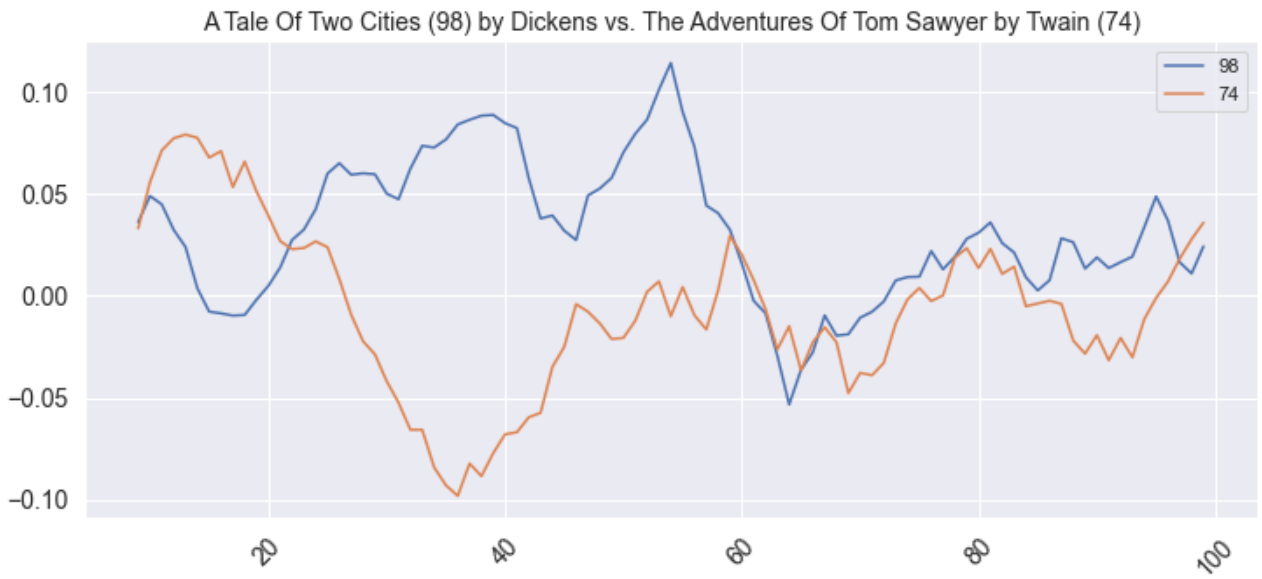




In [72]:
```
compare_novels(two_cities, huck)
```



In [73]:
```
compare_novels(two_cities, sawyer)
```



In [74]:
```
LIB.to_csv('full_LIB.csv')

CORPUS.to_csv('full_CORPUS.csv')
```

Sources

- `M10_04_AustenMelville.ipynb` by Professor Raf Alvarado
- Sentiment analysis using `VADER` in `nltk` : https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664
- How to fix matplotlib `.title()` TypeError: 'Text' object is not callable with `.set_title()` : https://techoverflow.net/2021/04/04/how-to-fix-matplotlib-title-typeerror-text-object-is-not-callable/
- Géron, Aurélien, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* (O'Reilly Media, 2019) (for plotting images)

In [ ]:
```

```