In [1]:

```
import pandas as pd
import numpy as np
import string
import re
from collections import Counter
from nltk.corpus import stopwords

pt1 = pd.read_csv('Shakespeare_works2.csv')

pt1.head()
```

Out[1]:

	Title	Publish Date	ParagraphNum	PlainText
0	All's Well That Ends Well	1602	1.0	Enter BERTRAM, the COUNTESS of Rousillon, HELE...
1	All's Well That Ends Well	1602	3.0	In delivering my son from me, I bury a second ...
2	All's Well That Ends Well	1602	4.0	And I in going, madam, weep o'er my father's d...
3	All's Well That Ends Well	1602	7.0	You shall find of the king a husband, madam; y...
4	All's Well That Ends Well	1602	12.0	What hope is there of his majesty's amendment?\n

In [2]:

```
pt1 = pt1[pt1.notnull()]
```

In [3]:

```
len(pt1.Title.unique())
```

Out[3]:

```
53
```

In [4]:

```
pt1.isnull().sum().sort_values(ascending = False)
```

Out[4]:

```
ParagraphNum    9
PlainText       9
Publish Date    6
Title           1
dtype: int64
```

In [5]:

```
pt1 = pt1.dropna()
```

In [6]:

```
pt1['Publish Date'] = pt1['Publish Date'].astype(int)
```

In [7]:

```
pt1.Title.unique()
```

Out[7]:

```
array(["All's Well That Ends Well", 'Antony and Cleopatra',
       'As You Like It', 'Comedy of Errors', 'Coriolanus', 'Cy
mbeline',
       'Hamlet', 'Henry IV, Part I', 'Henry IV, Part II', 'Hen
ry V',
       'Henry VI, Part I', 'Henry VI, Part II', 'Henry VI, Par
t III',
       'Henry VIII', 'Julius Caesar', 'King John', 'King Lea
r',
       "Lover's Complaint", "Love's Labour's Lost", 'Macbeth',
       'Measure for Measure', 'Merchant of Venice',
       'Merry Wives of Windsor', "Midsummer Night's Dream",
       'Much Ado about Nothing', 'Othello', 'Passionate Pilgri
m',
       'Pericles', 'Phoenix and the Turtle', 'Rape of Lucrec
e',
       'Richard II', 'Richard III', 'Romeo and Juliet', 'Sonne
ts',
       'Taming of the Shrew', '\nTaming of the Shrew"', 'Tempe
st',
       'Timon of Athens', 'Titus Andronicus', 'Troilus and Cre
ssida',
       'Twelfth Night', 'Two Gentlemen of Verona', 'Venus and
Adonis',
       "Winter's Tale"], dtype=object)
```

In [8]:

```
print(pt1['Publish Date'].min())
pt1['Publish Date'].max()
```

```
1589
```

Out[8]:

```
1612
```

In [9]:

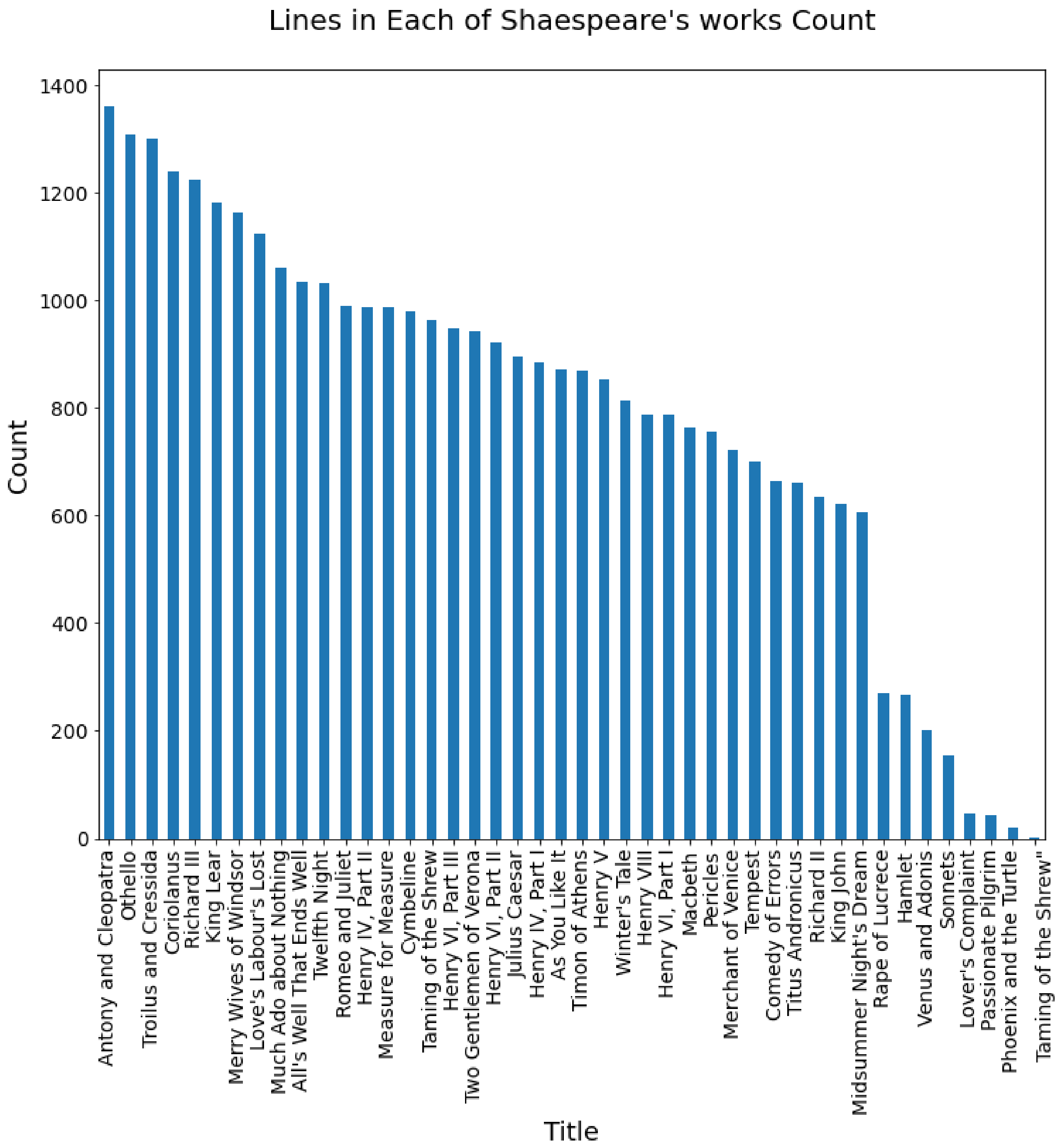
```
pt1['Publish Date'].value_counts().sort_index()
```

Out[9]:

```
1589     664
1590    1870
1591     787
1592    1224
1593    1829
1594    3324
1595    1241
1596    1343
1597    1871
1598    1958
1599    2798
1600    1430
1601    1320
1602    1034
1604    2296
1605    1946
1606    1361
1607    2110
1608     756
1609    1180
1610     814
1611     702
1612     788
Name: Publish Date, dtype: int64
```

In [10]:

```
ax = pt1['Title'].value_counts(ascending = False).plot(kind='ba
ax.set_title("Lines in Each of Shaespeare's works Count\n", fo
ax.set_xlabel('Title', fontsize=18)
ax.set_ylabel('Count', fontsize=18);
```



In [11]:
```
def clean_text(pt1):
    clean1 = re.sub(r'['+string.punctuation + '’—”'+']', "", p
    return re.sub(r'\W+', ' ', clean1)
```

In [12]:
```
pt1['tokenized'] = pt1['PlainText'].map(lambda x: clean_text(x
```

In [13]:
```
pt1['tokenized'].head()
```

Out[13]:

```
0    enter bertram the countess of rousillon helena...
1    in delivering my son from me i bury a second h...
2    and i in going madam weep oer my fathers death...
3    you shall find of the king a husband madam you...
4        what hope is there of his majestys amendment
Name: tokenized, dtype: object
```

In [14]:

```
pt1['num_wds'] = pt1['tokenized'].apply(lambda x: len(x.split(
pt1['num_wds'].mean()
```

Out[14]:

```
24.822519194134966
```

In [15]:

```
print(pt1['num_wds'].max())
pt1['num_wds'].min()
```

```
588
```

Out[15]:

```
1
```

In [16]:

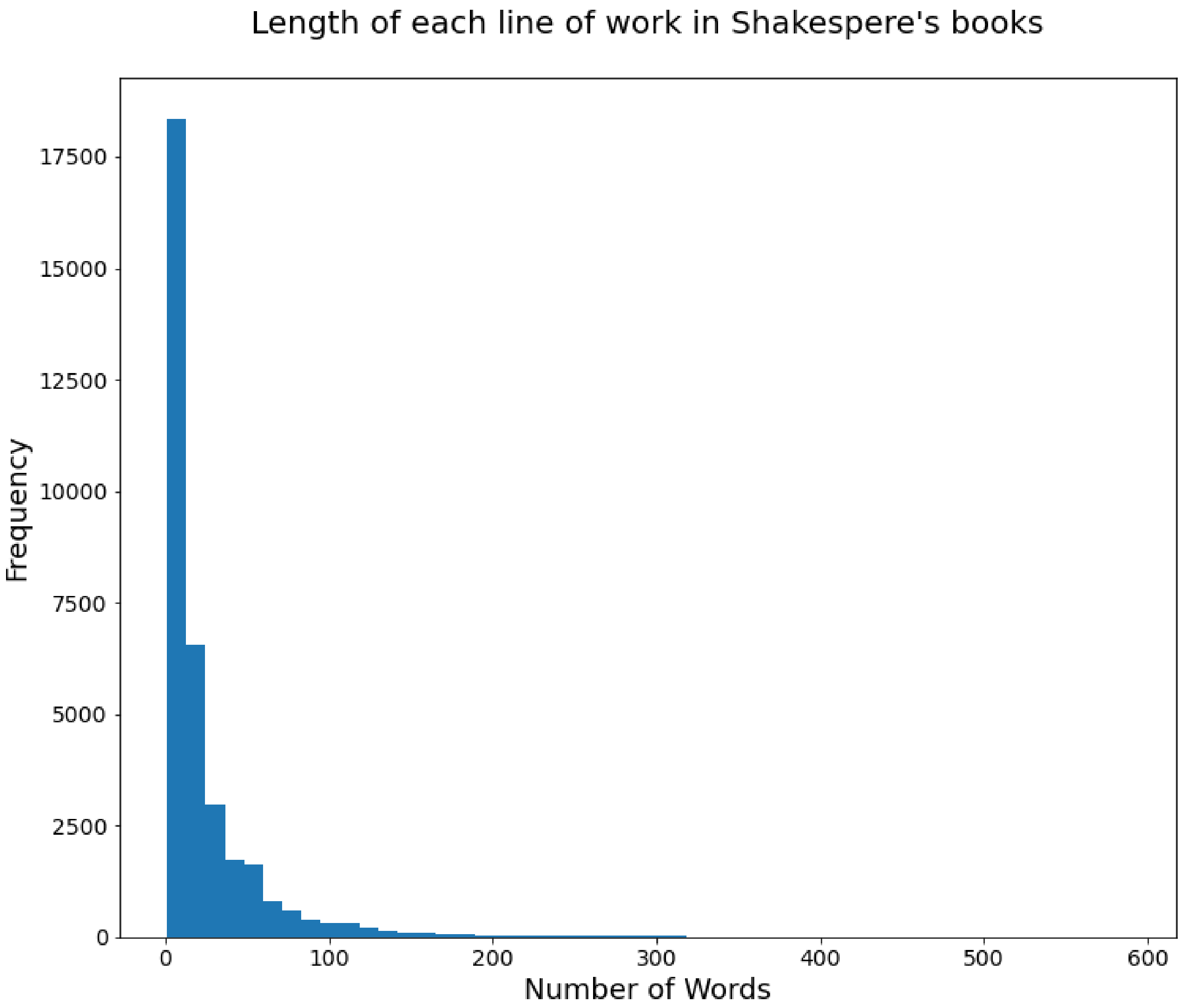
```
len(pt1[pt1['num_wds']==0])
```

Out[16]:

```
0
```

In [17]:

```
ax=pt1['num_wds'].plot(kind='hist', bins=50, fontsize=14, figs
ax.set_title("Length of each line of work in Shakespere's book
ax.set_ylabel('Frequency', fontsize=18)
ax.set_xlabel('Number of Words', fontsize=18);
```



In [18]:
```
pt1['uniq_wds'] = pt1['tokenized'].str.split().apply(lambda x:
pt1['uniq_wds'].head()
```

Out[18]:
```
0    12
1    11
2    24
3    40
4     8
Name: uniq_wds, dtype: int64
```

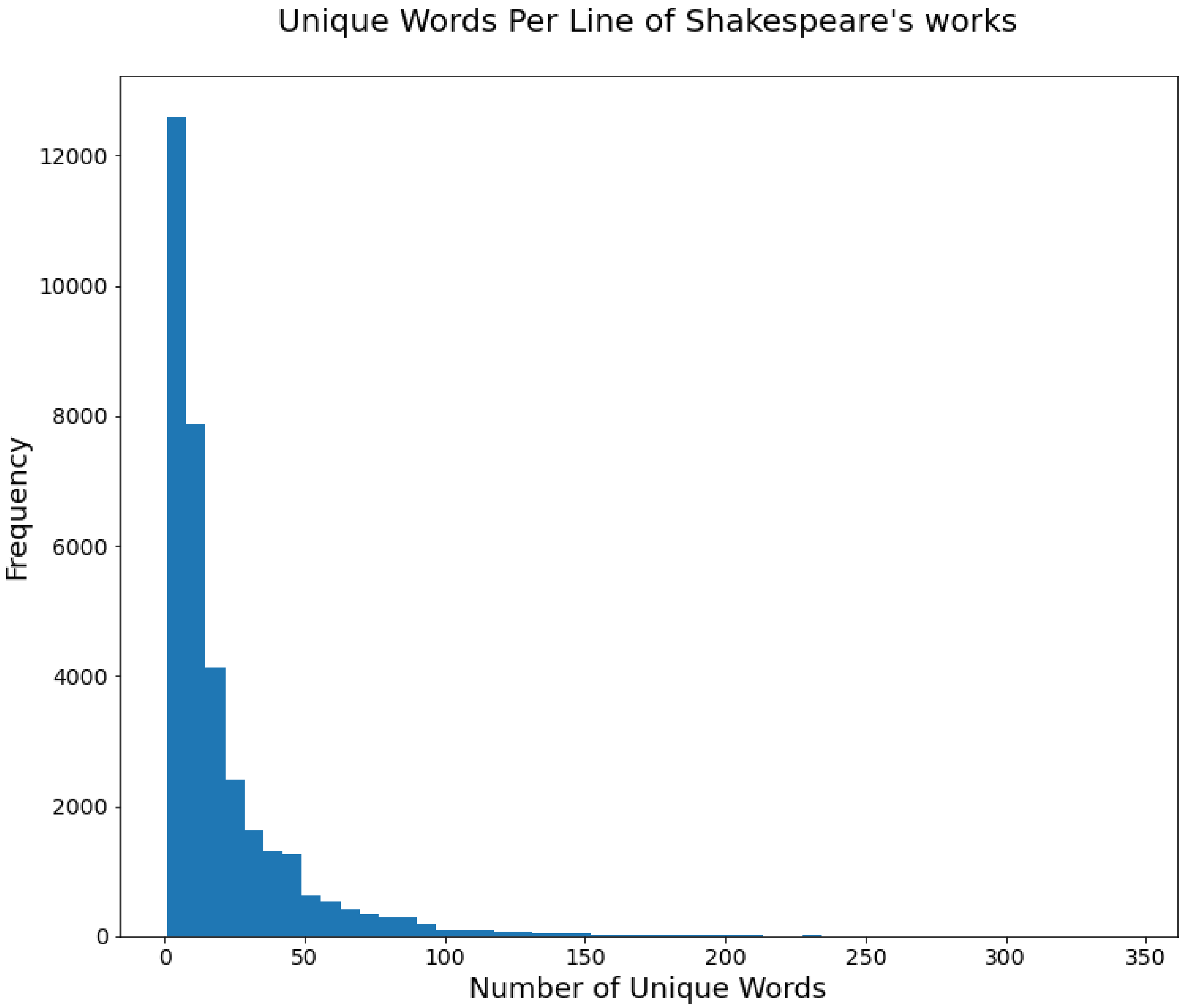
In [19]:
```
print(pt1['uniq_wds'].mean())
print(pt1['uniq_wds'].min())
pt1['uniq_wds'].max()
```

```
20.627980142007736
1
```

Out[19]:
```
344
```

In [20]:

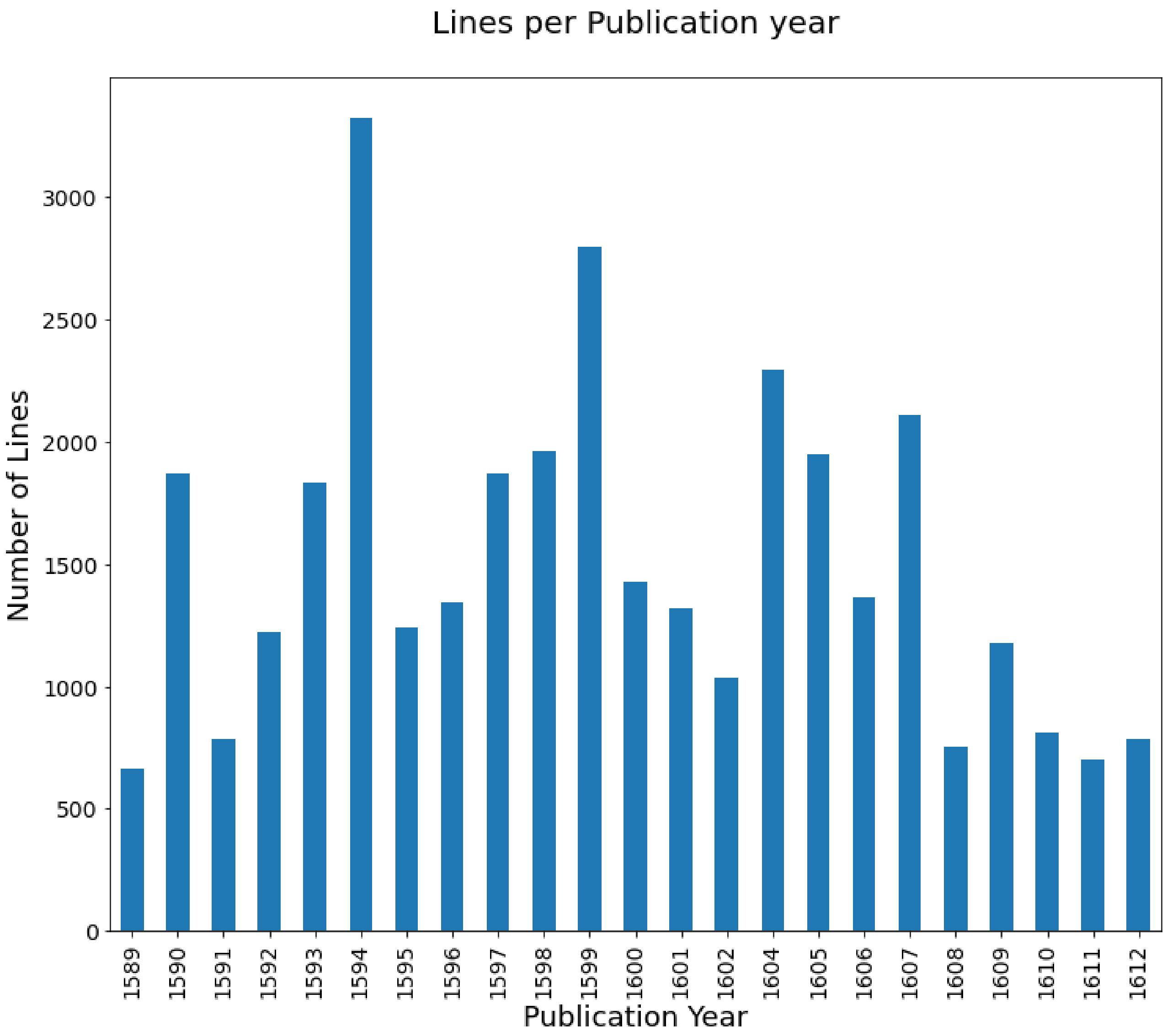
```
ax=pt1['uniq_wds'].plot(kind='hist', bins=50, fontsize=14, fig
ax.set_title("Unique Words Per Line of Shakespeare's works\n",
ax.set_ylabel('Frequency', fontsize=18)
ax.set_xlabel('Number of Unique Words', fontsize=18);
```



In [21]:

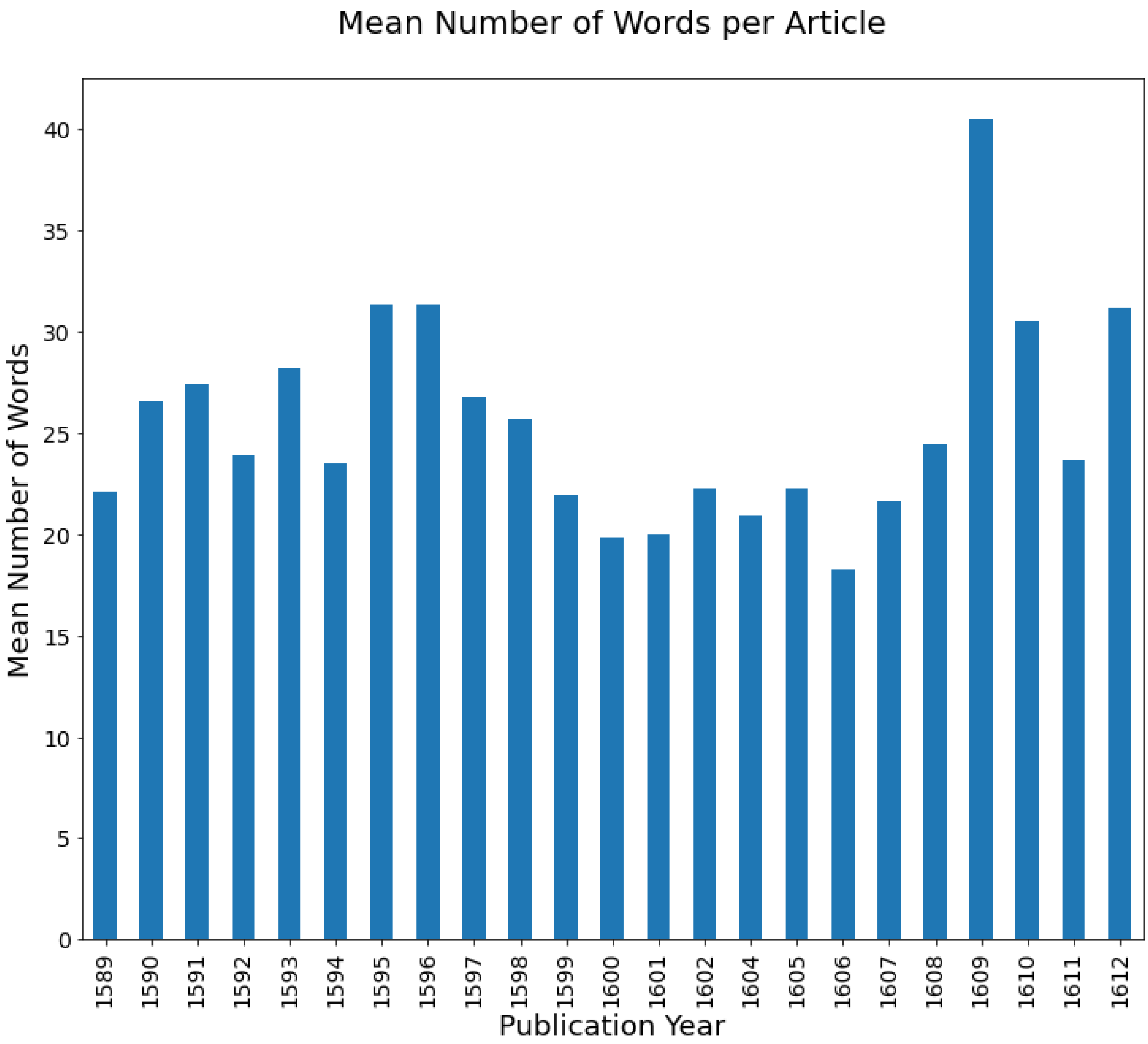
```
art_grps = pt1.groupby('Publish Date')

ax=art_grps['Publish Date'].aggregate(len).plot(kind='bar', fo
ax.set_title('Lines per Publication year\n', fontsize=20)
ax.set_ylabel('Number of Lines', fontsize=18)
ax.set_xlabel('Publication Year', fontsize=18);
```



In [22]:

```
ax=art_grps['num_wds'].aggregate(np.mean).plot(kind='bar', fon
ax.set_title('Mean Number of Words per Article\n', fontsize=20
ax.set_ylabel('Mean Number of Words', fontsize=18)
ax.set_xlabel('Publication Year', fontsize=18);
```



```
In [23]: wd_counts = Counter()
         for i, row in pt1.iterrows():
             wd_counts.update(row['tokenized'].split())
```

```
In [24]: wd_counts
```

```
Out[24]: Counter({'enter': 1725,
                  'bertram': 28,
                  'the': 24898,
                  'countess': 13,
                  'of': 15589,
                  'rousillon': 13,
                  'helena': 44,
                  'pand': 7072,
                  'lafeu': 18,
                  'all': 3663,
                  'in': 10185,
                  'black': 158,
                  'delivering': 3,
                  'my': 11639,
                  'son': 599,
                  'from': 2423,
                  'me': 7734,
                  'i': 17988,
                  'bury': 38,
                  'a': 13323,
                  'second': 105,
                  'husband': 278,
                  'and': 19458,
                  'going': 102,
                  'madam': 490,
                  'weep': 170,
                  'oer': 183,
                  'fathers': 267,
                  'death': 859,
                  'panew': 1,
                  'but': 4470,
                  'must': 1376,
                  'attend': 114,
                  'his': 6566,
                  'majestys': 13,
                  'command': 152,
                  'to': 16365,
                  'pwhom': 97,
                  'am': 2111,
                  'now': 2518,
                  'ward': 17,
                  'evermore': 23,
                  'subjection': 6,
                  'you': 12682,
                  'shall': 3220,
                  'find': 502,
                  'king': 1438,
                  'psir': 74,
                  'father': 803,
                  'he': 5742,
                  'that': 8452,
                  'so': 4539,
                  'generally': 10,
                  'is': 8615,
                  'at': 2373,
                  'times': 237,
                  'pgood': 168,
                  'necessity': 38,
                  'hold': 423,
                  'virtue': 177,
                  'whose': 414,
                  'pworthiness': 2,
                  'would': 2123,
                  'stir': 81,
                  'it': 7197,
                  'up': 1072,
                  'where': 920,
                  'wanted': 8,
                  'rather': 285,
                  'pthan': 451,
                  'lack': 94,
                  'there': 1545,
                  'such': 1295,
                  'abundance': 13,
                  'what': 3746,
                  'hope': 353,
                  'amendment': 4,
                  'hath': 1703,
                  'abandoned': 3,
                  'physicians': 10,
                  'under': 211,
                  'ppractises': 1,
                  'persecuted': 1,
                  'time': 1031,
                  'with': 7049,
                  'pfinds': 4,
                  'no': 3516,
                  'other': 647,
                  'advantage': 73,
                  'process': 17,
                  'only': 274,
                  'plosing': 6,
                  'by': 3345,
                  'this': 5988,
                  'young': 419,
                  'gentlewoman': 43,
                  'had': 1341,
                  'fathero': 1,
                  'phad': 146,
                  'how': 1678,
                  'sad': 174,
                  'passage': 36,
                  'tiswhose': 1,
                  'skill': 50,
                  'was': 2108,
                  'palmost': 19,
                  'as': 4527,
                  'great': 835,
                  'honesty': 76,
                  'stretched': 4,
                  'pfar': 11,
                  'have': 5503,
                  'made': 765,
                  'nature': 334,
                  'immortal': 27,
                  'pshould': 174,
                  'play': 281,
                  'for': 6094,
                  'work': 173,
                  'pkings': 7,
                  'sake': 180,
                  'were': 1446,
                  'living': 122,
                  'think': 957,
                  'be': 6615,
                  'pthe': 3182,
                  'kings': 263,
                  'disease': 27,
                  'called': 45,
                  'man': 1746,
                  'speak': 1064,
                  'famous': 28,
                  'sir': 2442,
                  'profession': 18,
                  'phis': 513,
                  'right': 339,
                  'gerard': 2,
                  'de': 135,
                  'narbon': 3,
                  'excellent': 105,
                  'indeed': 400,
                  'very': 742,
                  'plately': 3,
                  'spoke': 142,
                  'him': 5125,
                  'admiringly': 2,
                  'mourningly': 1,
                  'pwas': 185,
                  'skilful': 8,
                  'enough': 300,
                  'lived': 85,
                  'still': 561,
                  'if': 2648,
                  'knowledge': 74,
                  'pcould': 78,
                  'set': 432,
                  'against': 440,
                  'mortality': 14,
                  'good': 2629,
                  'lord': 2538,
                  'languishes': 2,
                  'fistula': 1,
                  'heard': 339,
                  'not': 8249,
                  'before': 632,
                  'notorious': 13,
                  'daughter': 397,
                  'sole': 24,
                  'child': 242,
                  'bequeathed': 3,
                  'poverlooking': 1,
                  'those': 518,
                  'hopes': 59,
                  'her': 4020,
                  'pher': 259,
                  'education': 9,
                  'promises': 22,
                  'dispositions': 6,
                  'she': 2265,
                  'pinherits': 1,
                  'which': 1495,
                  'makes': 322,
                  'fair': 793,
                  'gifts': 44,
                  'fairer': 38,
                  'pan': 144,
                  'unclean': 4,
                  'mind': 374,
                  'carries': 19,
                  'virtuous': 95,
                  'qualities': 24,
                  'pcommendations': 2,
                  'go': 1524,
                  'pity': 224,
                  'they': 2217,
                  'are': 3167,
                  'virtues': 59,
                  'ptraitors': 2,
                  'too': 1206,
                  'better': 568,
                  'their': 2062,
                  'psimpleness': 1,
                  'derives': 3,
                  'achieves': 1,
                  'goodness': 54,
                  'your': 6179,
                  'commendations': 12,
                  'get': 289,
                  'tears': 315,
                  'tis': 1153,
                  'best': 463,
                  'brine': 9,
                  'maiden': 44,
                  'can': 1146,
                  'season': 42,
                  'praise': 186,
                  'pin': 1030,
                  'remembrance': 58,
                  'never': 981,
                  'approaches': 14,
                  'heart': 1007,
                  'tyranny': 37,
                  'sorrows': 60,
                  'takes': 101,
                  'plivelihood': 1,
                  'cheek': 86,
                  'more': 2122,
                  'pgo': 177,
                  'lest': 78,
                  'thought': 376,
                  'affect': 23,
                  'pa': 1138,
                  'sorrow': 202,
                  'than': 1458,
                  'do': 3519,
                  'moderate': 8,
                  'lamentation': 8,
                  'dead': 548,
                  'pexcessive': 1,
                  'grief': 227,
                  'enemy': 161,
                  'excess': 15,
                  'pmakes': 48,
                  'soon': 151,
                  'mortal': 100,
                  'desire': 235,
                  'holy': 196,
                  'wishes': 31,
                  'understand': 88,
                  'we': 2909,
                  'thou': 5129,
                  'blest': 54,
                  'succeed': 15,
                  'thy': 3870,
                  'manners': 71,
                  'shape': 86,
                  'blood': 655,
                  'pcontend': 2,
                  'empire': 15,
                  'thee': 3284,
                  'pshare': 2,
                  'birthright': 4,
                  'love': 2057,
                  'trust': 169,
                  'few': 63,
                  'pdo': 249,
                  'wrong': 240,
                  'none': 459,
                  'able': 55,
                  'thine': 456,
                  'prather': 36,
                  'power': 341,
                  'use': 311,
                  'keep': 446,
                  'friend': 413,
                  'punder': 80,
                  'own': 767,
                  'lifes': 24,
                  'key': 29,
                  'chequed': 6,
                  'silence': 82,
                  'pbut': 1894,
                  'taxd': 4,
                  'speech': 115,
                  'heaven': 601,
                  'will': 4643,
                  'pthat': 2689,
                  'may': 1507,
                  'furnish': 18,
                  'prayers': 96,
                  'pluck': 96,
                  'down': 627,
                  'pfall': 20,
                  'on': 2882,
                  'head': 495,
                  'farewell': 301,
                  'ptis': 229,
                  'an': 1645,
                  'unseasond': 1,
                  'courtier': 24,
                  'padvise': 4,
                  'cannot': 705,
                  'want': 142,
                  'bless': 98,
                  'exit': 746,
                  'forged': 12,
                  'pyour': 483,
                  'thoughts': 249,
                  'servants': 88,
                  'comfortable': 13,
                  'pto': 3177,
                  'mother': 303,
                  'mistress': 443,
                  'make': 1541,
                  'much': 982,
                  'pretty': 121,
                  'lady': 655,
                  'credit': 53,
                  'exeunt': 936,
                  'o': 2087,
                  'these': 1179,
                  'grace': 571,
                  'shed': 50,
                  'like': 1597,
                  'pi': 2608,
                  'forgot': 86,
                  'imagination': 27,
                  'pcarries': 5,
                  'favour': 121,
                  'int': 71,
                  'bertrams': 1,
                  'undone': 54,
                  'pif': 925,
                  'away': 822,
                  'twere': 100,
                  'one': 1706,
                  'should': 1429,
                  'bright': 80,
                  'particular': 48,
                  'star': 43,
                  'wed': 33,
                  'above': 130,
                  'radiance': 3,
                  'collateral': 1,
                  'light': 292,
                  'pmust': 140,
                  'comforted': 6,
                  'sphere': 9,
                  'ambition': 38,
                  'thus': 696,
                  'plagues': 14,
                  'itself': 238,
                  'hind': 10,
                  'mated': 7,
                  'lion': 92,
                  'die': 481,
                  'twas': 132,
                  'though': 448,
                  'plague': 91,
                  'see': 1383,
                  'every': 551,
                  'hour': 303,
                  'sit': 204,
                  'draw': 190,
                  'arched': 2,
                  'brows': 48,
                  'hawking': 4,
                  'eye': 463,
                  'curls': 3,
                  'our': 2795,
                  'hearts': 224,
                  'table': 47,
                  'capable': 13,
                  'pof': 1088,
                  'line': 42,
                  'trick': 43,
                  'sweet': 770,
                  'hes': 268,
                  'gone': 468,
                  'idolatrous': 1,
                  'fancy': 47,
                  'sanctify': 4,
                  'reliques': 2,
                  'who': 811,
                  'comes': 578,
                  'here': 1921,
                  'penter': 316,
                  'parolles': 31,
                  'paside': 59,
                  'pone': 158,
                  'goes': 163,
                  'yet': 1331,
                  'know': 1600,
                  'liar': 12,
                  'pthink': 64,
                  'way': 559,
                  'fool': 380,
                  'solely': 8,
                  'coward': 87,
                  'pyet': 308,
                  'fixed': 19,
                  'evils': 26,
                  'fit': 149,
                  'take': 1081,
                  'place': 416,
                  'when': 1462,
                  'steely': 2,
                  'bones': 74,
                  'plook': 122,
                  'bleak': 6,
                  'cold': 197,
                  'wind': 186,
                  'withal': 132,
                  'full': 401,
                  'oft': 139,
                  'pcold': 11,
                  'wisdom': 89,
                  'waiting': 10,
                  'superfluous': 17,
                  'folly': 80,
                  'save': 172,
                  'queen': 486,
                  'monarch': 16,
                  'meditating': 5,
                  'virginity': 22,
                  'ay': 694,
                  'some': 1174,
                  'stain': 45,
                  'soldier': 138,
                  'let': 1716,
                  'pask': 8,
                  'question': 123,
                  'pmay': 164,
                  'barricado': 2,
                  'out': 1255,
                  'assails': 3,
                  'valiant': 141,
                  'defence': 37,
                  'weak': 112,
                  'unfold': 31,
                  'us': 1611,
                  'pwarlike': 4,
                  'resistance': 4,
                  'sitting': 22,
                  'pundermine': 2,
                  'blow': 92,
                  'poor': 612,
                  'underminers': 1,
                  'pblowers': 1,
                  'military': 8,
                  'policy': 44,
                  'pvirgins': 2,
                  'might': 464,
                  'men': 858,
                  'being': 602,
                  'blown': 36,
                  'quicklier': 1,
                  'pblown': 3,
                  'marry': 332,
                  'blowing': 7,
                  'again': 763,
                  'breach': 35,
                  'yourselves': 71,
                  'lose': 203,
                  'city': 129,
                  'pis': 628,
                  'politic': 10,
                  'commonwealth': 26,
                  'ppreserve': 3,
                  'loss': 128,
                  'rational': 2,
                  'pincrease': 2,
                  'virgin': 35,
                  'got': 118,
                  'till': 436,
                  'pvirginity': 5,
                  'first': 508,
                  'lost': 262,
                  'pmetal': 1,
                  'virgins': 10,
                  'once': 399,
                  'ten': 137,
                  'found': 210,
                  'ever': 604,
                  'kept': 101,
                  'pever': 23,
                  'companion': 36,
                  't': 119,
                  'stand': 523,
                  'little': 482,
                  'therefore': 461,
                  'theres': 296,
                  'said': 385,
                  'prule': 2,
                  'part': 480,
                  'accuse': 33,
                  'mothers': 96,
                  'most': 1063,
                  'infallible': 5,
                  'pdisobedience': 1,
                  'hangs': 37,
                  'himself': 439,
                  'murders': 21,
                  'buried': 53,
                  'phighways': 1,
                  'sanctified': 7,
                  'limit': 19,
                  'desperate': 59,
                  'poffendress': 1,
                  'breeds': 25,
                  'mites': 1,
                  'pmuch': 42,
                  'cheese': 12,
                  'consumes': 2,
                  'pparing': 1,
                  'dies': 98,
                  'feeding': 15,
                  'stomach': 42,
                  'pbesides': 42,
                  'peevish': 29,
                  'proud': 206,
                  'idle': 70,
                  'pselflove': 3,
                  'inhibited': 2,
                  'sin': 176,
                  'pcanon': 1,
                  'choose': 94,
                  'loose': 44,
                  'pbyt': 1,
                  'within': 419,
                  'year': 84,
                  'pitself': 7,
                  'goodly': 79,
                  'increase': 30,
                  'pprincipal': 1,
                  'worse': 160,
                  'liking': 29,
                  'ill': 1572,
                  'neer': 219,
                  'plikes': 1,
                  'commodity': 18,
                  'gloss': 15,
                  'plying': 4,
                  'longer': 111,
                  'less': 208,
                  'worth': 209,
                  'off': 468,
                  'pwhile': 73,
                  'vendible': 2,
                  'answer': 369,
                  'request': 62,
                  'old': 624,
                  'wears': 34,
                  'cap': 45,
                  'fashion': 91,
                  'richly': 13,
                  'suited': 8,
                  'unsuitable': 1,
                  'just': 140,
                  'plike': 221,
                  'brooch': 5,
                  'toothpick': 2,
                  'wear': 182,
                  'pnow': 330,
                  'date': 22,
                  'pie': 8,
                  'pporridge': 2,
                  'french': 149,
                  'pwithered': 1,
                  'pears': 7,
                  'looks': 222,
                  'eats': 16,
                  'drily': 1,
                  'withered': 8,
                  'pear': 8,
                  'formerly': 6,
                  'pmarry': 25,
                  'anything': 22,
                  'pthere': 269,
                  'master': 740,
                  'thousand': 317,
                  'loves': 289,
                  'phoenix': 17,
                  'captain': 136,
                  'guide': 27,
                  'goddess': 29,
                  'sovereign': 145,
                  'counsellor': 13,
                  'traitress': 1,
                  'dear': 416,
                  'humble': 64,
                  'humility': 16,
                  'jarring': 4,
                  'concord': 9,
                  'discord': 17,
                  'dulcet': 6,
                  'faith': 400,
                  'disaster': 6,
                  'world': 615,
                  'fond': 62,
                  'adoptious': 1,
                  'christendoms': 1,
                  'blinking': 2,
                  'cupid': 34,
                  'gossips': 9,
                  'god': 695,
                  'send': 227,
                  'well': 2315,
                  'courts': 10,
                  'learning': 29,
                  'wish': 221,
                  'whats': 297,
                  'wishing': 7,
                  'body': 264,
                  'pwhich': 944,
                  'felt': 40,
                  'poorer': 5,
                  'born': 168,
                  'pwhose': 238,
                  'baser': 11,
                  'stars': 77,
                  'shut': 56,
                  'pmight': 47,
                  'effects': 26,
                  'them': 1973,
                  'follow': 280,
                  'friends': 466,
                  'show': 397,
                  'alone': 227,
                  'preturn': 19,
                  'thanks': 157,
                  'page': 201,
                  'monsieur': 40,
                  'calls': 92,
                  'helen': 49,
                  'remember': 169,
                  'pwill': 370,
                  'court': 227,
                  'charitable': 17,
                  'mars': 42,
                  'especially': 12,
                  'why': 1214,
                  'wars': 145,
                  'needs': 139,
                  'pbe': 362,
                  'predominant': 5,
                  'retrograde': 2,
                  'backward': 20,
                  'fight': 271,
                  'thats': 370,
                  'running': 26,
                  'fear': 637,
                  'proposes': 1,
                  'safety': 68,
                  'composition': 15,
                  'valour': 98,
                  'wing': 25,
                  'businesses': 5,
                  'pacutely': 1,
                  'return': 190,
                  'perfect': 55,
                  'instruction': 15,
                  'serve': 183,
                  'naturalize': 1,
                  'pthee': 50,
                  'wilt': 287,
                  'courtiers': 13,
                  'pcounsel': 5,
                  'advice': 43,
                  'thrust': 48,
                  'upon': 1482,
                  'else': 387,
                  'diest': 21,
                  'unthankfulness': 4,
                  'pthine': 21,
                  'ignorance': 35,
                  'pthou': 568,
                  'hast': 558,
                  'leisure': 61,
                  'say': 1575,
                  'pnone': 29,
                  'uses': 13,
                  'remedies': 8,
                  'ourselves': 97,
                  'lie': 309,
                  'ascribe': 2,
                  'fated': 4,
                  'sky': 47,
                  'pgives': 23,
                  'free': 175,
                  'scope': 28,
                  'doth': 965,
                  'pull': 10,
                  'pour': 320,
                  'slow': 54,
                  'designs': 11,
                  'dull': 91,
                  'pwhat': 742,
                  'mounts': 7,
                  'high': 237,
                  'feed': 80,
                  'mine': 1142,
                  'mightiest': 7,
                  'space': 32,
                  'fortune': 297,
                  'brings': 51,
                  'join': 49,
                  'likes': 23,
                  'kiss': 209,
                  'native': 39,
                  'things': 309,
                  'pimpossible': 5,
                  'strange': 233,
                  'attempts': 6,
                  'weigh': 36,
                  'pains': 87,
                  'sense': 120,
                  'suppose': 28,
                  'been': 686,
                  'strove': 3,
                  'pso': 641,
                  'merit': 47,
                  'did': 1519,
                  'miss': 23,
                  'diseasemy': 1,
                  'project': 10,
                  'deceive': 25,
                  'intents': 22,
                  'fixd': 24,
                  'leave': 626,
                  'flourish': 109,
                  'cornets': 14,
                  'france': 366,
                  'pwith': 1017,
                  'letters': 117,
                  'divers': 21,
                  'attendants': 109,
                  'florentines': 2,
                  'senoys': 1,
                  'ears': 171,
                  'phave': 426,
                  'fought': 65,
                  'equal': 47,
                  'continue': 28,
                  'braving': 4,
                  'war': 254,
                  'reported': 14,
                  'nay': 506,
                  'credible': 1,
                  'received': 69,
                  'certainty': 5,
                  'vouchd': 3,
                  'cousin': 237,
                  'austria': 6,
                  'caution': 6,
                  'florentine': 9,
                  'move': 94,
                  'pfor': 1654,
                  'speedy': 20,
                  'aid': 63,
                  'wherein': 76,
                  'dearest': 54,
                  'pprejudicates': 1,
                  'business': 221,
                  'seem': 181,
                  'denial': 11,
                  'papproved': 1,
                  'majesty': 245,
                  'plead': 55,
                  'amplest': 2,
                  'credence': 3,
                  'armd': 38,
                  'florence': 13,
                  'denied': 54,
                  'gentlemen': 187,
                  'mean': 283,
                  'tuscan': 2,
                  'service': 210,
                  'freely': 51,
                  'either': 123,
                  'nursery': 5,
                  'gentry': 13,
                  'sick': 161,
                  'breathing': 21,
                  'exploit': 14,
                  'count': 97,
                  'pyoung': 30,
                  'youth': 262,
                  'bearst': 10,
                  'face': 452,
                  'pfrank': 1,
                  'curious': 16,
                  'haste': 161,
                  'phath': 280,
                  'composed': 10,
                  'moral': 27,
                  'parts': 114,
                  'pmayst': 3,
                  'inherit': 15,
                  'welcome': 335,
                  'paris': 89,
                  'duty': 149,
                  'corporal': 21,
                  'soundness': 1,
                  'pas': 1316,
                  'myself': 544,
                  'friendship': 39,
                  'pfirst': 39,
                  'tried': 22,
                  'soldiership': 8,
                  'look': 717,
                  'far': 278,
                  'pinto': 105,
                  'pdiscipled': 1,
                  'bravest': 4,
                  'lasted': 2,
                  'long': 446,
                  'both': 554,
                  'haggish': 1,
                  'age': 203,
                  'steal': 82,
                  'wore': 24,
                  'act': 117,
                  'repairs': 1,
                  'talk': 192,
                  'phe': 740,
                  'wit': 263,
                  'observe': 28,
                  'ptoday': 15,
                  'lords': 459,
                  'jest': 103,
                  'ptill': 214,
                  'scorn': 103,
                  'unnoted': 3,
                  'pere': 72,
                  'hide': 107,
                  'levity': 5,
                  'honour': 625,
                  'contempt': 45,
                  'nor': 711,
                  'bitterness': 8,
                  'pwere': 191,
                  'pride': 128,
                  'or': 1805,
                  'sharpness': 3,
                  'awaked': 12,
                  'pclock': 6,
                  'knew': 147,
                  'true': 788,
                  'minute': 38,
                  'pexception': 1,
                  'bid': 312,
                  'tongue': 406,
                  'obeyd': 8,
                  'hand': 863,
                  'below': 52,
                  'used': 69,
                  'creatures': 42,
                  'another': 333,
                  'bowd': 15,
                  'eminent': 5,
                  'top': 47,
                  'low': 100,
                  'ranks': 20,
                  'pmaking': 34,
                  'humbled': 9,
                  'copy': 12,
                  'younger': 32,
                  'followd': 40,
                  'demonstrate': 5,
                  'goers': 1,
                  'plies': 22,
                  'richer': 14,
                  'tomb': 56,
                  'approof': 4,
                  'lives': 196,
                  'epitaph': 12,
                  'royal': 215,
                  'always': 59,
                  'pmethinks': 43,
                  'hear': 853,
                  'plausive': 3,
                  'words': 417,
                  'scatterd': 12,
                  'grafted': 5,
                  'grow': 135,
                  'bearlet': 1,
                  'live': 521,
                  'pthis': 673,
                  'melancholy': 65,
                  'began': 39,
                  'pon': 209,
                  'catastrophe': 4,
                  'heel': 14,
                  'pastime': 11,
                  'pwhen': 660,
                  'outlet': 1,
                  'quoth': 114,
                  'pafter': 74,
                  'flame': 24,
                  'lacks': 16,
                  'oil': 13,
                  'snuff': 7,
                  'spirits': 122,
                  'apprehensive': 2,
                  'senses': 38,
                  'pall': 254,
                  'new': 198,
                  'disdain': 39,
                  'judgments': 10,
                  'pmere': 2,
                  'garments': 44,
                  'constancies': 1,
                  'pexpire': 2,
                  'fashions': 6,
                  'wishd': 30,
                  'after': 341,
                  'psince': 137,
                  'wax': 26,
                  'honey': 35,
                  'bring': 418,
                  'home': 323,
                  'quickly': 125,
                  'dissolved': 7,
                  'hive': 9,
                  'give': 1172,
                  'labourers': 1,
                  'room': 43,
                  'loved': 156,
                  'pthey': 316,
                  'least': 101,
                  'lend': 91,
                  'fill': 69,
                  'knowt': 19,
                  'ist': 144,
                  'physician': 22,
                  'died': 107,
                  'famed': 8,
                  'six': 48,
                  'months': 39,
                  'since': 325,
                  'try': 82,
                  'plend': 15,
                  'arm': 138,
                  'rest': 342,
                  'worn': 33,
                  'several': 80,
                  'applications': 1,
                  'sickness': 50,
                  'pdebate': 2,
                  'pmy': 907,
                  'sons': 168,
                  'dearer': 23,
                  'thank': 283,
                  'steward': 29,
                  'clown': 41,
                  'care': 220,
                  'even': 410,
                  'content': 156,
                  'pwish': 6,
                  'calendar': 8,
                  'past': 145,
                  'pendeavours': 1,
                  'then': 1959,
                  'wound': 82,
                  'modesty': 47,
                  'pfoul': 10,
                  'clearness': 3,
                  'deservings': 6,
                  'pourselves': 7,
                  'publish': 9,
                  'does': 276,
                  'knave': 156,
                  'sirrah': 128,
                  'complaints': 9,
                  'pbelieve': 26,
                  'slowness': 2,
                  'pyou': 828,
                  'commit': 32,
                  'ability': 7,
                  'penough': 16,
                  'knaveries': 4,
                  'yours': 249,
                  'unknown': 42,
                  'fellow': 280,
                  'pmany': 32,
                  'rich': 164,
                  'damned': 58,
                  'ladyships': 6,
                  'isbel': 2,
                  'woman': 311,
                  'beggar': 52,
                  'beg': 90,
                  'case': 103,
                  'isbels': 3,
                  'pheritage': 1,
                  'pblessing': 5,
                  'issue': 112,
                  'barnes': 2,
                  'blessings': 19,
                  ...})
```

In [25]:

```
for sw in stopwords.words('english'):
    del wd_counts[sw]
```

In [26]:

```
wd_counts.most_common(20)
```

Out[26]:

```
[('pand', 7072),
 ('thou', 5129),
 ('thy', 3870),
 ('thee', 3284),
 ('shall', 3220),
 ('pthe', 3182),
 ('pto', 3177),
 ('pthat', 2689),
 ('good', 2629),
 ('pi', 2608),
 ('lord', 2538),
 ('sir', 2442),
 ('well', 2315),
 ('come', 2186),
 ('would', 2123),
 ('love', 2057),
 ('pbut', 1894),
 ('man', 1746),
 ('enter', 1725),
 ('let', 1716)]
```

In [27]:

```
# Recognize and count dimensia words: sad, remember, forgot, o
def dementia_count(pt1):
    dementia = ['sad', 'remember', 'forgot', 'old', 'this worl
                'forgetful', 'forgetfulness', 'forgetting', 'f
                'forgetting appointments', 'forgetting where y
                'Alzheimer', 'Alzheimer\'s', 'Alzheimer\'s dis
                'Alzheimer\'', 'mental deterioration', 'mental
                'mental loss', 'softening of the brain', 'soft
    dementia_count = 0
    for word in dementia:
        if word in pt1:
            dementia_count += 1
    return dementia_count
```

In [28]:

```
pt1['num_dim_words'] = pt1.tokenized.apply(dementia_count)
```

In [29]:

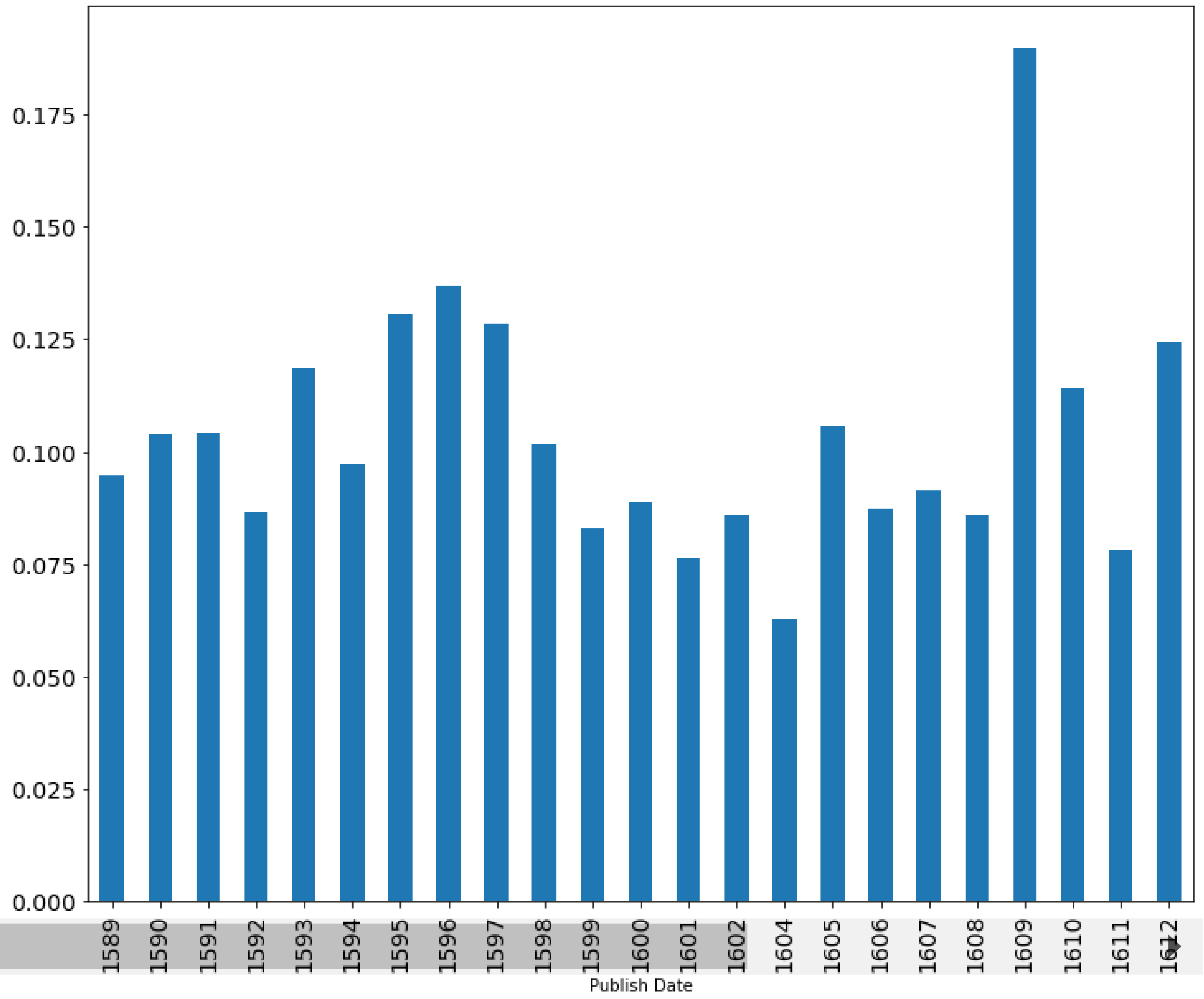
```
pt1
```

Out[29]:

	Title	Publish Date	ParagraphNum	PlainText	tokenized	nu
0	All's Well That Ends Well	1602	1.0	Enter BERTRAM, the COUNTESS of Rousillon, HELE...	enter bertram the countess of rousillon helena...	
1	All's Well That Ends Well	1602	3.0	In delivering my son from me, I bury a second ...	in delivering my son from me i bury a second h...	
2	All's Well That Ends Well	1602	4.0	And I in going, madam, weep o'er my father's d...	and i in going madam weep oer my fathers death...	
3	All's Well That Ends Well	1602	7.0	You shall find of the king a husband, madam; y...	you shall find of the king a husband madam you...	
4	All's Well That Ends Well	1602	12.0	What hope is there of his majesty's amendment?\n	what hope is there of his majestys amendment	
...	...	...	...	...	...	
34650	Winter's Tale	1610	3430.0	That she is living,\n[p]Were it but told you, ...	that she is living pwere it but told you shoul...	
34651	Winter's Tale	1610	3437.0	You gods, look down\n[p]And	you gods look down pand from	

	Title	Publish Date	ParagraphNum	PlainText	tokenized	nu
				from your sacred v...	your sacred vials...	
34652	Winter's Tale	1610	3445.0	There's time enough for that;\n[p]Lest they de...	theres time enough for that plest they desire ...	
34653	Winter's Tale	1610	3453.0	O, peace, Paulina!\n[p]Thou shouldst a husband...	o peace paulina pthou shouldst a husband take ...	
34654	Winter's Tale	1610	3474.0	[Exeunt]	exeunt	

In [30]:
```
# Chart the number of dementia words per line per year
ax=art_grps['num_dim_words'].aggregate(np.mean).plot(kind='bar
```



In [31]:

```
import requests
import json
import os

# Check if the word is in the local repository of definitions 
def check_definition(word):
    try:
        file = open('word_definitions.csv', 'r')
        file.close()
    except:
        file = open('word_definitions.csv', 'w+')
        if os.stat("word_definitions.csv").st_size == 0:
            file.write('word,definition')
        file.close()

    try:
        #if exits in local repository, return the definition
        word_definitions = pd.read_csv('word_definitions.csv',
        if word in word_definitions:
            return word_definitions[word_definitions["word"]==w
        #else, get the definition from the API
        else:
            definition = get_definition(word)
            #add the word and definition to the local reposito
            word_definitions = word_definitions.append({'word'
            # print(word_definitions, word, definition, " inte
            word_definitions.to_csv('word_definitions.csv')
            return definition
    except:
        word_definitions = pd.read_csv('word_definitions.csv',
        definition = ""
        #add the word and definition to the local repository
        word_definitions = word_definitions.append({'word':wor
        word_definitions.to_csv('word_definitions.csv')
        return definition


# Get the definition of a word from the API
def get_definition(word):
    word = word.lower()
    response = ''
#     717d065b-80fb-4a21-9aae-3ddbb7a5a2de
#     c7b1669c-7629-42a4-befd-3f32b966aa74

    base_url = "https://www.dictionaryapi.com/api/v3/reference
    api = "c7b1669c-7629-42a4-befd-3f32b966aa74"


    api_key = "?key=" + api


    full_api = base_url + word + api_key
    try:

        response = requests.get(full_api)
        json_data = json.loads(response.text)
        definition = json_data[0]["shortdef"][0]
        definition = re.sub(' +', ' ', definition)
        definition = re.sub(',', '', definition)
        definition = re.sub("\'", '', definition)
        definition = re.sub('\[', '', definition)
        definition = re.sub('\]', '', definition)
        definition = re.sub('\{', '', definition)
        definition = re.sub('\}', '', definition)
        definition = re.sub('\"', '', definition)
        definition = re.sub('h:', '', definition)
        definition = re.sub('https', '', definition)
        definition = re.sub('http', '', definition)
        definition = re.sub('www', '', definition)
        definition = re.sub('\.', '', definition)
        definition = re.sub('\:', '', definition)
        definition = re.sub('\;', '', definition)
        definition = re.sub('\?', '', definition)
        definition = re.sub('\!', '', definition)
        definition = re.sub('\(', '', definition)
        definition = re.sub('\)', '', definition)
        definition = re.sub('\*', '', definition)
        definition = re.sub('\&', '', definition)
        definition = re.sub('\%', '', definition)
        definition = re.sub('\$', '', definition)
        definition = re.sub('\#', '', definition)
        definition = re.sub('\@', '', definition)
        definition = re.sub('\^', '', definition)
        definition = re.sub('\+', '', definition)
        definition = re.sub('\=', '', definition)
        definition = re.sub('\-', '', definition)
        definition = re.sub('\_', '', definition)
        definition = re.sub('\|', '', definition)
        definition = re.sub('\~', '', definition)
        definition = re.sub('\`', '', definition)
        definition = re.sub('\>', '', definition)
        definition = re.sub('\<', '', definition)
        definition = re.sub('\/', '', definition)
        return definition
    except:
        return ''
```

In [32]:

```
def word_complex(wordlist):
    stop_words = set(stopwords.words('english'))

    word_depth_value = 0

    known_words = set()
    known_words.add(wordlist)

    unknown_words = set()
    unknown_words2ndLine = set()
    try:
        wordlist = re.sub("[^a-zA-Z]", "", wordlist)
        word_list = wordlist.split(" ")
        for word in word_list:
            word_definition = check_definition(word)
            if (word_definition == -1):
                print('This code can\'t be run without an API 
                return;

            word_definition_arr = word_definition.split(" ")
            for word in word_definition_arr:
                if word not in stop_words and len(word) > 1:
                    #print("Adding word: " + str(word))
                    unknown_words.add(word)

            while len(unknown_words) > 0:
                word = unknown_words.pop()
                known_words.add(word)
                word = re.sub("[^a-zA-Z]", "", word)
                word_definition = check_definition(word)
                try:
                    word_definition_arr = word_definition.spli
                except:
                    continue

                for word in word_definition_arr:
                    if word not in known_words and word not in
                        unknown_words2ndLine.add(word)
                        word_depth_value += 1

                if word_depth_value % 50 is 0:
                    pass
                    # print("NUM UNKNOWN WORDS: " + str(len(un
                    # print("NUM KNOWN WORDS: " + str(len(know

                # print("Now I know " + str(word_depth_value) 


            for word in unknown_words2ndLine:
                    if word not in known_words and word not in
                        unknown_words.add(word)

            # print("I needed to learn " + str(word_depth_valu
            return word_depth_value
    except:
        wordlist = re.sub("[^a-zA-Z]", "", wordlist)
        word_definition = check_definition(wordlist)
        if (word_definition == -1):
            print('This code can\'t be run without an API key!'
            return;
        word_definition_arr = word_definition.split(" ")
        for word in word_definition_arr:
            if word not in stop_words and len(word) > 1:
                #print("Adding word: " + str(word))
                unknown_words.add(word)

        while len(unknown_words) > 0:
            word = unknown_words.pop()
            known_words.add(word)
            word = re.sub("[^a-zA-Z]", "", word)
            word_definition = check_definition(word)
            try:
                word_definition_arr = word_definition.split("
            except:
                continue

            for word in word_definition_arr:
                if word not in known_words and word not in unkı
                    unknown_words2ndLine.add(word)
                    word_depth_value += 1

            if word_depth_value % 50 is 0:
                pass

        for word in unknown_words2ndLine:
                if word not in known_words and word not in unkı
                    unknown_words.add(word)

        # print("I needed to learn " + str(word_depth_value) +
        return word_depth_value

def word_complex2(wordlist):
    stop_words = set(stopwords.words('english'))

    word_depth_value = 0

    known_words = set()
    known_words.add(wordlist)

    unknown_words = set()
    unknown_words2ndLine = set()
    try:
        word_list = wordlist.split(" ")
        initial_complexity = 0
        total_complexity = 0
        for word in word_list:
            word = re.sub("[^a-zA-Z]", "", word)
            try:
                initial_complexity = check_complex(word)
                total_complexity += initial_complexity
            except:
                word_definition = check_definition(word)
                if (word_definition == -1):
                    print('This code can\'t be run without an
                    return;

                word_definition_arr = word_definition.split("
                for word in word_definition_arr:
                    if word not in stop_words and len(word) >
                        #print("Adding word: " + str(word))
                        unknown_words.add(word)

                while len(unknown_words) > 0:
                    word = unknown_words.pop()
                    known_words.add(word)
                    word = re.sub("[^a-zA-Z]", "", word)
                    word_definition = check_definition(word)
                    try:
                        word_definition_arr = word_definition.
                    except:
                        continue

                    for word in word_definition_arr:
                        if word not in known_words and word no
                            unknown_words2ndLine.add(word)
                            word_depth_value += 1

                    if word_depth_value % 50 is 0:
                        pass
                        # print("NUM UNKNOWN WORDS: " + str(le
                        # print("NUM KNOWN WORDS: " + str(len(
                    # print("Now I know " + str(word_depth_valu


                for word in unknown_words2ndLine:
                        if word not in known_words and word no
                            unknown_words.add(word)

                # print("I needed to learn " + str(word_depth_
                return word_depth_value
        return total_complexity
    except:
        word = re.sub("[^a-zA-Z]", "", wordlist)
        try:
            return check_complex(word)
        except:
            word_definition = check_definition(word)
            if (word_definition == -1):
                print('This code can\'t be run without an API
                return;
            word_definition_arr = word_definition.split(" ")
            for word in word_definition_arr:
                if word not in stop_words and len(word) > 1:
                    #print("Adding word: " + str(word))
                    unknown_words.add(word)

            while len(unknown_words) > 0:
                word = unknown_words.pop()
                known_words.add(word)
                word = re.sub("[^a-zA-Z]", "", word)
                word_definition = check_definition(word)
                try:
                    word_definition_arr = word_definition.spli
                except:
                    continue

                for word in word_definition_arr:
                    if word not in known_words and word not in
                        unknown_words2ndLine.add(word)
                        word_depth_value += 1

                if word_depth_value % 50 is 0:
                    pass
                    # print("NUM UNKNOWN WORDS: " + str(len(un
                    # print("NUM KNOWN WORDS: " + str(len(known

                # print("Now I know " + str(word_depth_value) 


            for word in unknown_words2ndLine:
                    if word not in known_words and word not in
                        unknown_words.add(word)

            # print("I needed to learn " + str(word_depth_value
            return word_depth_value


def check_complex(word):
    try:
        file = open('word_complex.csv', 'r')
        file.close()
    except:
        file = open('word_complex.csv', 'w+')
        if os.stat("word_complex.csv").st_size == 0:
            file.write('word,complexity')
        file.close()

    try:
        #if exits in local repository, return the definition
        word_complexity = pd.read_csv('word_complex.csv', inde
        if word in word_complexity:
            return word_complexity[word_complexity["word"]==wo
        #else, get the definition from the API
        else:
            complexity = word_complex(word)
            #add the word and definition to the local reposito
            word_complexity = word_complexity.append({'word':wo
            # print(word_definitions, word, definition, " inter
            word_complexity.to_csv('word_complex.csv')
            return complexity
    except:
        word_complexity = pd.read_csv('word_complex.csv', inde
        complexity = 0
        #add the word and definition to the local repository
        word_complexity = word_complexity.append({'word':word,
        word_complexity.to_csv('word_complex.csv')
        return complexity
```

```
<>:41: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:82: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:138: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:183: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:41: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:82: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:138: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:183: SyntaxWarning: "is" with a literal. Did you mean "=="?
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:41: SyntaxWarning: "is" with a literal. Did you mean "=="?
  if word_depth_value % 50 is 0:
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:82: SyntaxWarning: "is" with a literal. Did you mean "=="?
  if word_depth_value % 50 is 0:
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:138: SyntaxWarning: "is" with a literal. Did you mean "=="?
  if word_depth_value % 50 is 0:
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:183: SyntaxWarning: "is" with a literal. Did you mean "=="?
  if word_depth_value % 50 is 0:
```

In [33]:

```
word_complex2("""did the barber shave the barber""")
# test word: love in the word_complex function
```

```
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:218: FutureWarning: The frame.append method is deprecated an
d will be removed from pandas in a future version. Use pandas.
concat instead.
  word_complexity = word_complexity.append({'word':word, 'comp
lexity':complexity}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
```

```
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:218: FutureWarning: The frame.append method is deprecated an
d will be removed from pandas in a future version. Use pandas.
```

```
concat instead.
  word_complexity = word_complexity.append({'word':word, 'comp
lexity':complexity}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:218: FutureWarning: The frame.append method is deprecated an
d will be removed from pandas in a future version. Use pandas.
concat instead.
  word_complexity = word_complexity.append({'word':word, 'comp
lexity':complexity}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
```

```
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:218: FutureWarning: The frame.append method is deprecated an
d will be removed from pandas in a future version. Use pandas.
concat instead.
  word_complexity = word_complexity.append({'word':word, 'comp
lexity':complexity}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
```

```
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:218: FutureWarning: The frame.append method is deprecated an
d will be removed from pandas in a future version. Use pandas.
concat instead.
  word_complexity = word_complexity.append({'word':word, 'comp
lexity':complexity}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
```

```
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
```

```
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\3500028379.p
y:25: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.co
ncat instead.
  word_definitions = word_definitions.append({'word':word, 'de
finition':definition}, ignore_index=True)
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1007951961.p
y:218: FutureWarning: The frame.append method is deprecated an
d will be removed from pandas in a future version. Use pandas.
concat instead.
  word_complexity = word_complexity.append({'word':word, 'comp
lexity':complexity}, ignore_index=True)
```

Out[33]:

```
303
```

In [34]:

```python
#calculation of word_complex of pt1.PlainText and store values
if os.path.exists('pt1.csv'):
    pt1 = pd.read_csv('pt1.csv', index_col=0)
else:
    pt1['word_complexity'] = pt1['PlainText'].apply(word_compl
pt1
```

Out[34]:

	Title	Publish Date	ParagraphNum	PlainText	tokenized	nu
0	All's Well That Ends Well	1602	1.0	Enter BERTRAM, the COUNTESS of Rousillon, HELE...	enter bertram the countess of rousillon helena...	
1	All's Well That Ends Well	1602	3.0	In delivering my son from me, I bury a second ...	in delivering my son from me i bury a second h...	
2	All's Well That Ends Well	1602	4.0	And I in going, madam, weep o'er my father's d...	and i in going madam weep oer my fathers death...	
3	All's Well That Ends Well	1602	7.0	You shall find of the king a husband, madam; y...	you shall find of the king a husband madam you...	
4	All's Well That Ends Well	1602	12.0	What hope is there of his majesty's amendment?\n	what hope is there of his majestys amendment	
...	...	...	...	...	...	
34650	Winter's Tale	1610	3430.0	That she is living,\n[p]Were it but told you, ...	that she is living pwere it but told you shoul...	
34651	Winter's Tale	1610	3437.0	You gods, look down\n[p]And	you gods look down pand from	

	Title	Publish Date	ParagraphNum	PlainText	tokenized	nu
				from your sacred v...	your sacred vials...	
34652	Winter's Tale	1610	3445.0	There's time enough for that;\n[p]Lest they de...	theres time enough for that plest they desire ...	
34653	Winter's Tale	1610	3453.0	O, peace, Paulina!\n[p]Thou shouldst a husband...	o peace paulina pthou shouldst a husband take ...	
34654	Winter's Tale	1610	3474.0	[Exeunt]	exeunt	

In [35]:

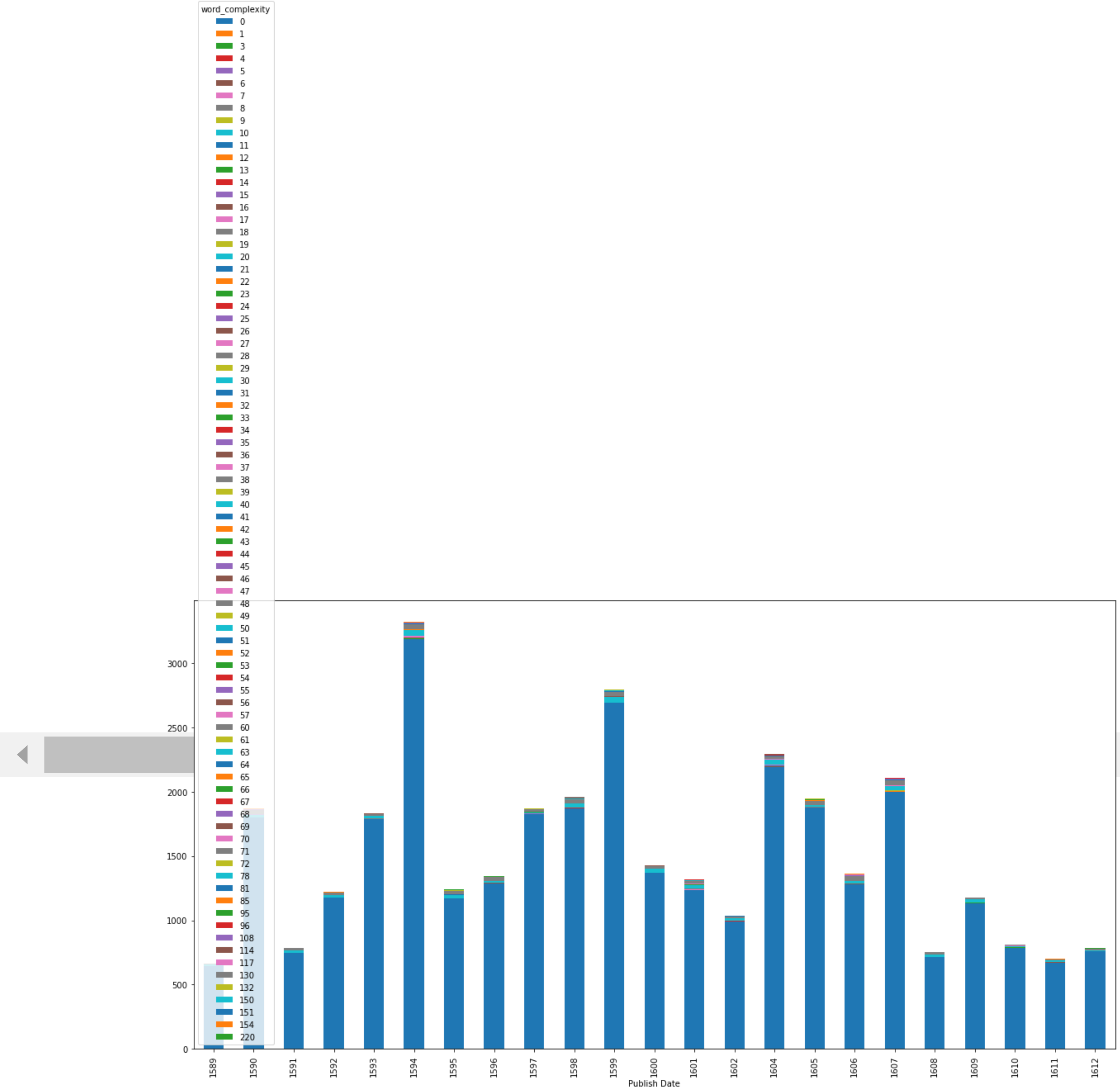
```
pt1.to_csv('pt1.csv')
# stores a copy of the dataframe in a csv file
```

In [36]:

```
# Chart the number of pt1.words_complexity per line per year
pt1.groupby(['Publish Date', 'word_complexity']).size().unstac
```

Out[36]:

```
<AxesSubplot:xlabel='Publish Date'>
```

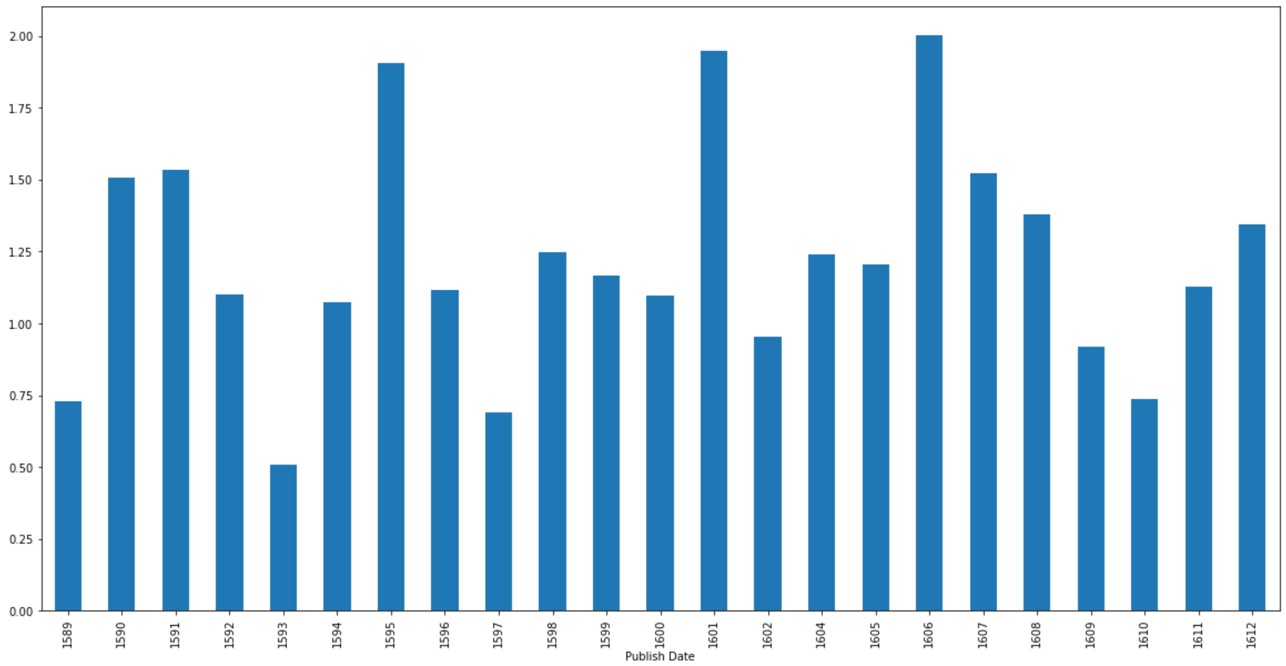


In [37]:

```
# Chart the number of mean word_complexity per line per year
pt1.groupby(['Publish Date'])['word_complexity'].mean().plot(k
```

Out[37]:

```
<AxesSubplot:xlabel='Publish Date'>
```



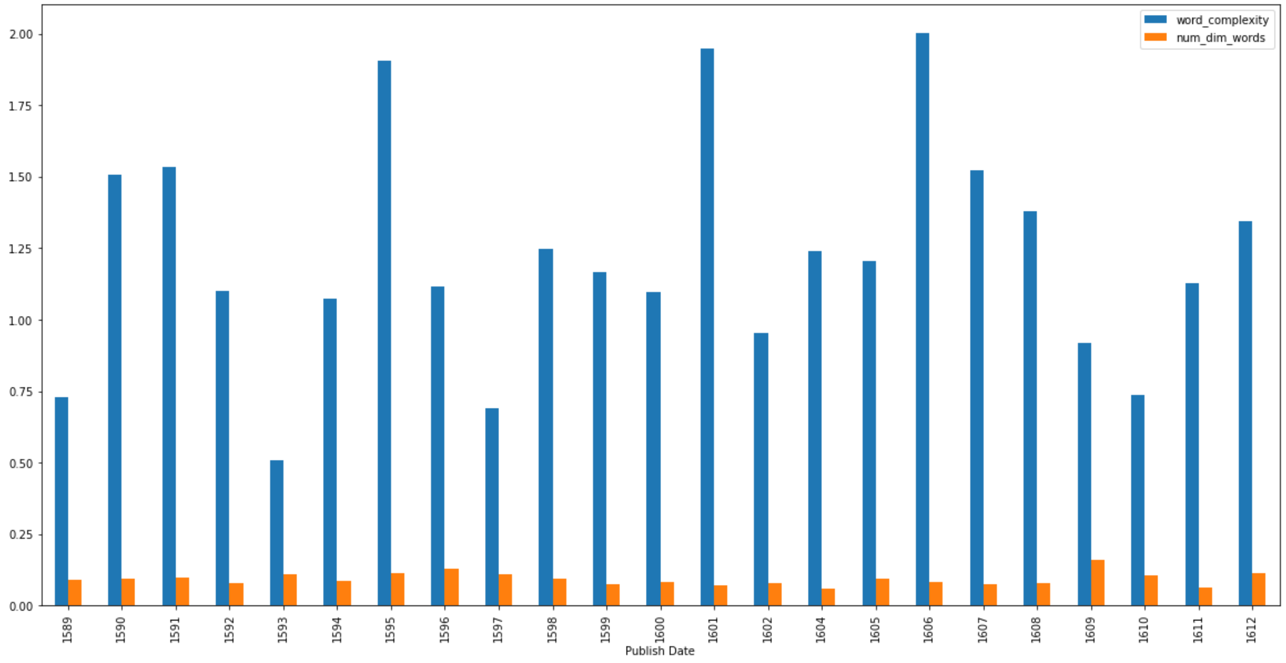
In [38]:

```
# Chart the number of mean word_complexity and num_dim_words pe
pt1.groupby(['Publish Date'])['word_complexity', 'num_dim_word
```

```
C:\Users\theoj\AppData\Local\Temp\ipykernel_38456\1499341372.p
y:2: FutureWarning: Indexing with multiple keys (implicitly co
nverted to a tuple of keys) will be deprecated, use a list ins
tead.
  pt1.groupby(['Publish Date'])['word_complexity', 'num_dim_wo
rds'].mean().plot(kind='bar', figsize=(20,10))
```

Out[38]:

```
<AxesSubplot:xlabel='Publish Date'>
```



In [39]:

```
# create new column for the mean word_complexity and num_dim_wo
pt1['mean_word_complexity'] = pt1.groupby(['Publish Date'])['wo
# create new column for the mean num_dim_words per line per yea
pt1['mean_num_dim_words'] = pt1.groupby(['Publish Date'])['num_
#create new column for the product of mean_word_complexity and
pt1['mean_dim_word_complexity'] = pt1['mean_word_complexity']*
pt1
```

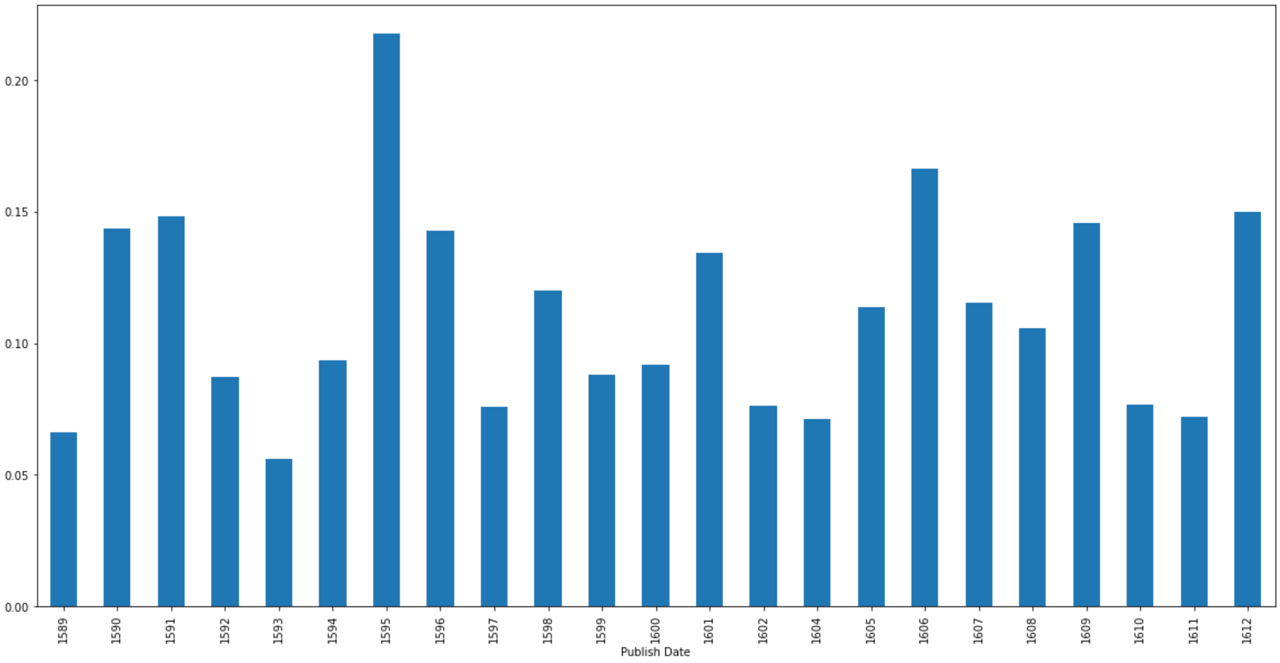
Out[39]:

	Title	Publish Date	ParagraphNum	PlainText	tokenized	nu
0	All's Well That Ends Well	1602	1.0	Enter BERTRAM, the COUNTESS of Rousillon, HELE...	enter bertram the countess of rousillon helena...	
1	All's Well That Ends Well	1602	3.0	In delivering my son from me, I bury a second ...	in delivering my son from me i bury a second h...	
2	All's Well That Ends Well	1602	4.0	And I in going, madam, weep o'er my father's d...	and i in going madam weep oer my fathers death...	
3	All's Well That Ends Well	1602	7.0	You shall find of the king a husband, madam; y...	you shall find of the king a husband madam you...	
4	All's Well That Ends Well	1602	12.0	What hope is there of his majesty's amendment?\n	what hope is there of his majestys amendment	
...	...	...	...	...	...	
34650	Winter's Tale	1610	3430.0	That she is living,\n[p]Were it but told you, ...	that she is living pwere it but told you shoul...	
34651	Winter's Tale	1610	3437.0	You gods, look down\n[p]And	you gods look down pand from	

	Title	Publish Date	ParagraphNum	PlainText	tokenized	nu
				from your sacred v...	your sacred vials...	
34652	Winter's Tale	1610	3445.0	There's time enough for that;\n[p]Lest they de...	theres time enough for that plest they desire ...	
34653	Winter's Tale	1610	3453.0	O, peace, Paulina!\n[p]Thou shouldst a husband...	o peace paulina pthou shouldst a husband take ...	
34654	Winter's Tale	1610	3474.0	[Exeunt]	exeunt	

```
In [40]: # Chart the mean_dim_word_complexity per line per year
         pt1.groupby(['Publish Date'])['mean_dim_word_complexity'].mean
```

```
Out[40]: <AxesSubplot:xlabel='Publish Date'>
```


Publish Date