Big Data Project - Analysis of song lyrics

September 6, 2018

1 Advisory Regarding Innapropriate and Offencive Words

Some genres of music more than others feature profanity and words that others might find offensive. While every effort has been made to obfuscate such offensive words in the notebook, it might be necessary to use the real *word in the code and it's inclusion in this notebook is purely for research purposes.

- Usage of the F-Word and N-Word

2 IMPORTS

In [1]:
```
import os
from requests import session
import zipfile
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
from sklearn import datasets
import scipy.stats as stats
import pylab
from PIL import Image, ImageDraw, ImageFilter, ImageFont
from wordcloud import WordCloud

%matplotlib inline
```

```
/Users/robertgray/anaconda3/envs/py36/lib/python3.6/site-packages/statsmodels/compat/pandas.py
  from pandas.core import datetools
```

3 CONSTANTS

In [2]:
```
PROJECT_DIRECTORY = os.getcwd()
DATA_DIRECTORY = os.path.join(PROJECT_DIRECTORY,'data')

DATA_SQLLITE = os.path.join(DATA_DIRECTORY, "summary.db")
```

4 Connect to SQLLITE DB

```
In [3]: import sqlite3

        conn = sqlite3.connect(DATA_SQLLITE)
```

4.1 SQL Queries

```
In [10]: def getTopWordsByGenre(genre,count):
             SQL = """
             SELECT
                 word,
                 SUM(count) AS count
             FROM
                 summary_word_counts
             WHERE
                 genre NOT IN ('GENRE','OTHER','NOT AVAILABLE')
                 AND LENGTH(word) > 3
                 AND genre = '""" + genre + """'
             GROUP BY
                 genre,
                 word
             ORDER BY
                 count desc
             LIMIT """ + count + """;"""
             df = pd.read_sql_query(SQL, conn)
             return df;

         def getProfanityCountByGenre():
             SQL = """
             SELECT
                 year,
                 SUM(count) AS count
             FROM
                 profanity_summary
             GROUP BY
                 year
             ORDER BY
                 count DESC"""
             df = pd.read_sql_query(SQL, conn)
             return df;

         def getProfanityCountsAndTotals():
             SQL = """
             WITH RECURSIVE totals AS (
                 SELECT
                     genre,
                     SUM(count) AS total_word_count
```

```
        FROM
            summary_word_counts
        WHERE
            genre NOT IN ('GENRE','OTHER','NOT AVAILABLE')
            AND YEAR NOT IN ('702','67','112')
        GROUP BY
            genre
        ORDER BY
            genre ASC
    ),
    profanity AS (
        SELECT
            genre AS genre,
            SUM(count) AS count
        FROM
            profanity_summary
        WHERE
            genre NOT IN ('GENRE','OTHER','NOT AVAILABLE')
            AND YEAR NOT IN ('702','67','112')
        GROUP BY
            genre
        ORDER BY
            genre ASC
    )
    SELECT
        totals.genre,
        totals.total_word_count,
        profanity.count
    FROM
        totals,
        profanity
    WHERE
        totals.genre = profanity.genre
        """
    df = pd.read_sql_query(SQL, conn)
    return df;

def getProfanityByYear(genre):
    SQL = """
    WITH RECURSIVE totals AS (
    SELECT
        "year",
        SUM(count) AS total_word_count
    FROM
        summary_word_counts
    WHERE
        genre = '""" + genre + """'
        AND YEAR NOT IN ('702','67','112')
```

```
    GROUP BY
        "year"
    ORDER BY
        "year" ASC
    ),
    profanity AS (
    SELECT
        "year" AS "year",
        SUM(count) AS count
    FROM
        profanity_summary
    WHERE
        genre = '""" + genre + """'
        AND YEAR NOT IN ('702','67','112')
    GROUP BY
        year
    ORDER BY
        year ASC
    )
    SELECT
        totals."YEAR",
        totals.total_word_count,
        profanity.count
    FROM
        totals,
        profanity
    WHERE
        totals."YEAR" = profanity."YEAR"
    """
    df = pd.read_sql_query(SQL, conn)
    return df;

def getWordByYear(word):
    SQL = """
    WITH RECURSIVE word AS (
    SELECT
        "year",
        SUM(count) AS word_count
    FROM
        summary_word_counts
    WHERE
        word LIKE '""" + word + """%'
        AND genre NOT IN ('GENRE','OTHER','NOT AVAILABLE')
        AND YEAR NOT IN ('702','67','112')
    GROUP BY
        "year"
    ORDER BY
        "year" ASC
```

```
), totals AS (
SELECT
    "year",
    SUM(count) AS total_word_count
FROM
    summary_word_counts
WHERE
    genre NOT IN ('GENRE','OTHER','NOT AVAILABLE')
    AND YEAR NOT IN ('702','67','112')
GROUP BY
    "year"
ORDER BY
    "year" ASC
)
SELECT
    word."YEAR",
    word.word_count,
    totals.total_word_count
FROM
    word,
    totals
WHERE
    word."YEAR" = totals."YEAR"
"""
df = pd.read_sql_query(SQL, conn)
return df;
```

5 Most used words in Hip-Hop

```
In [26]: df_hiphop = getTopWordsByGenre('HIP-HOP','100')
         df_hiphop.replace('NIG*', 'N-WORD',inplace=True,regex=True)
         df_hiphop.replace('FUCK*', 'F-WORD',inplace=True,regex=True)
         df_hiphop.head(100)
```

Out[26]:

	word	count
0	LIKE	99097
1	KNOW	62234
2	JUST	49676
3	N-WORDA	43616
4	CAUSE	39455
5	SHIT	37278
6	DOWN	33585
7	YEAH	33444
8	LOVE	32174
9	BACK	31643
10	N-WORDAS	30444
11	MAKE	29538

12	F-WORD	29359
13	WANT	27244
14	COME	25853
15	TIME	25280
16	BABY	25240
17	NEVER	24918
18	THEM	24511
19	SOME	23678
20	BITCH	22933
21	TAKE	21831
22	RIGHT	21740
23	WANNA	21613
24	GIRL	21487
25	THEN	21072
26	MONEY	20597
27	ABOUT	19126
28	TELL	18958
29	KEEP	18776
..	...	...
70	STAY	8775
71	LIVE	8564
72	N-WORDAZ	8485
73	PEOPLE	8228
74	HIGH	8078
75	NAME	7966
76	EVER	7895
77	BLACK	7875
78	RIDE	7801
79	F-WORDIN	7768
80	MADE	7672
81	BEFORE	7646
82	WATCH	7597
83	TURN	7570
84	HOLD	7566
85	THING	7559
86	LEAVE	7514
87	FACE	7508
88	ALWAYS	7354
89	AWAY	7291
90	SAME	7283
91	HEAR	7265
92	DONE	7253
93	ROCK	7234
94	WHILE	7217
95	DAMN	7188
96	WELL	7060
97	LONG	6982
98	AGAIN	6920

```
99      TALK   6914

[100 rows x 2 columns]
```

```
In [30]: d = {}
         for a, x in df_hiphop.values:
             d[a] = x

         import matplotlib.pyplot as plt
         from wordcloud import WordCloud
         from matplotlib.backends.backend_pdf import PdfPages
         with PdfPages('WordCloud.pdf') as pdf_pages:
             wordcloud = WordCloud()
             wordcloud.generate_from_frequencies(frequencies=d)
             plt.figure(num=None, figsize=(10, 10), dpi=80, facecolor='w', edgecolor='k')
             plt.imshow(wordcloud, interpolation="bilinear")
             plt.axis("off")
             pdf_pages.savefig()
             plt.show()
```

6 Which genres feature the most amount of inappropriate lyrics?

```
In [6]: data = getProfanityCountsAndTotals()
        data.set_index("genre",drop=True,inplace=True)
        data['percentage'] = (((data['count']) / (data['total_word_count'])) * 100)
```

```
In [7]: data.head(10)
```

Out[7]:

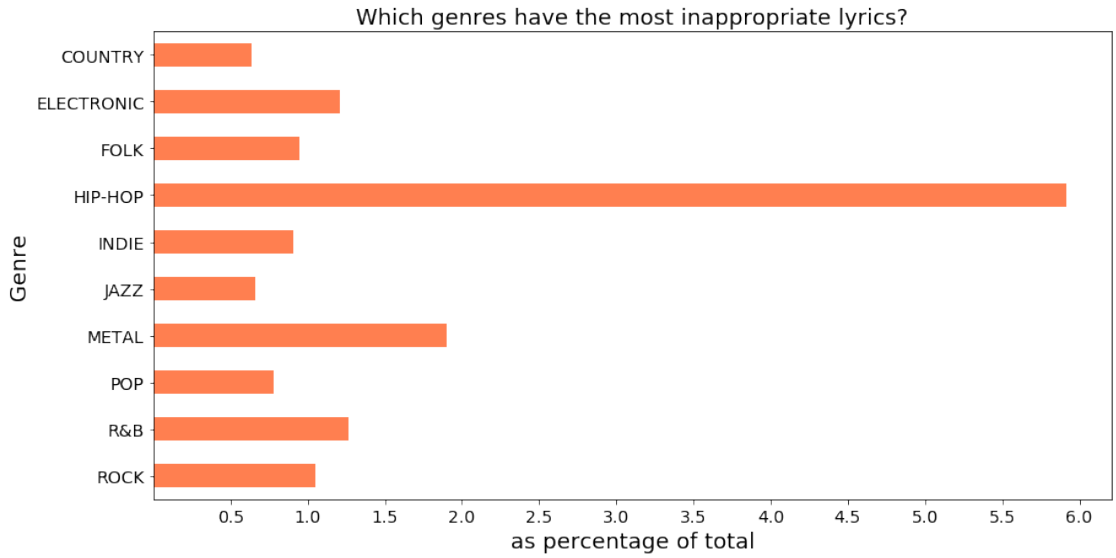
genre	total_word_count	count	percentage
COUNTRY	1223528	7792	0.636847
ELECTRONIC	694240	8400	1.209956
FOLK	191480	1818	0.949446
HIP-HOP	5633020	333248	5.915974
INDIE	279104	2525	0.904681
JAZZ	623406	4098	0.657357
METAL	2012263	38289	1.902783
POP	4388710	34262	0.780685
R&B	335055	4248	1.267852
ROCK	9527605	100094	1.050568

```
In [8]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10 entries, COUNTRY to ROCK
Data columns (total 3 columns):
total_word_count    10 non-null int64
count               10 non-null int64
percentage          10 non-null float64
dtypes: float64(1), int64(2)
memory usage: 320.0+ bytes
```

```
In [9]: data.to_csv(DATA_DIRECTORY+ "InappropriateLyricsByGenre.csv", sep=",", index=False)
```

```
In [10]: from matplotlib.backends.backend_pdf import PdfPages
         with PdfPages('InappropriateLyricsByGenre.pdf') as pdf_pages:
             plt.figure()
             ax = data['percentage'].plot(kind='barh', figsize=(14,7),
                                          color="coral", fontsize=14);
             ax.set_alpha(0.8)
             ax.set_title("Which genres have the most inappropriate lyrics?", fontsize=18)
             ax.set_xlabel("as percentage of total", fontsize=18);
             ax.set_xticks([0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0])
             ax.set_ylabel("Genre", fontsize=18);
             # invert
             ax.invert_yaxis()
             pdf_pages.savefig(ax.figure)
```



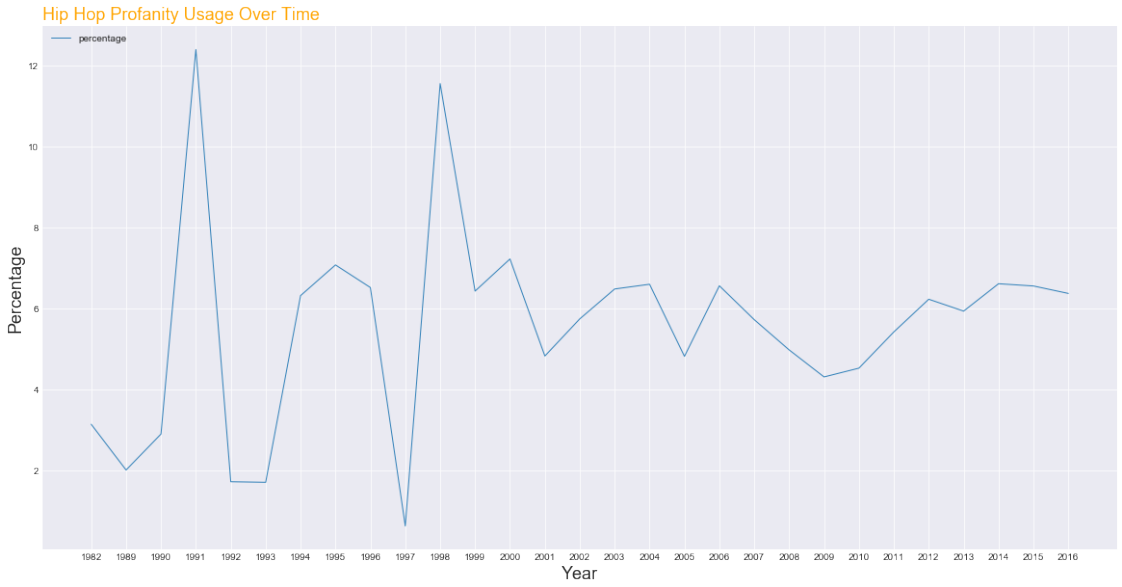

```
In [11]: hipHopByYear = getProfanityByYear('HIP-HOP')
         hipHopByYear['percentage'] = (((hipHopByYear['count']) / (hipHopByYear['total_word_cou
         hipHopByYear.head(200)
```

Out[11]:

	year	total_word_count	count	percentage
0	1982	414	13	3.140097
1	1989	7511	151	2.010385
2	1990	1035	30	2.898551
3	1991	1606	199	12.391034
4	1992	10053	173	1.720879
5	1993	3047	52	1.706597
6	1994	10464	661	6.316896
7	1995	18166	1285	7.073654
8	1996	19336	1260	6.516343
9	1997	1272	8	0.628931
10	1998	8873	1025	11.551899
11	1999	26027	1673	6.427940
12	2000	20358	1470	7.220749
13	2001	19917	961	4.825024
14	2002	46153	2650	5.741772
15	2003	27268	1767	6.480123
16	2004	105538	6962	6.596676
17	2005	131275	6323	4.816606
18	2006	1535984	100743	6.558857
19	2007	939708	53770	5.721990
20	2008	367635	18292	4.975587
21	2009	265508	11441	4.309098
22	2010	324034	14665	4.525760
23	2011	287256	15570	5.420252

```
24  2012            352940   21976    6.226554
25  2013            269109   15968    5.933655
26  2014            288661   19080    6.609830
27  2015            235805   15455    6.554144
28  2016            308067   19625    6.370367
```

In [12]:
```
from matplotlib.backends.backend_pdf import PdfPages
with PdfPages('InappropriateHIP-HOPLyricsByYear.pdf') as pdf_pages:
    plt.figure(figsize=(20, 10))
    plt.style.use('seaborn-darkgrid')
    plt.plot('year', 'percentage', data=hipHopByYear,linewidth=1, alpha=0.9, label='p
    plt.legend(loc=2, ncol=2)
    plt.title("Hip Hop Profanity Usage Over Time", loc='left', fontsize=18, fontweigh
    plt.xlabel("Year", fontsize=18)
    plt.ylabel("Percentage", fontsize=18)
    pdf_pages.savefig()
    plt.show()
```



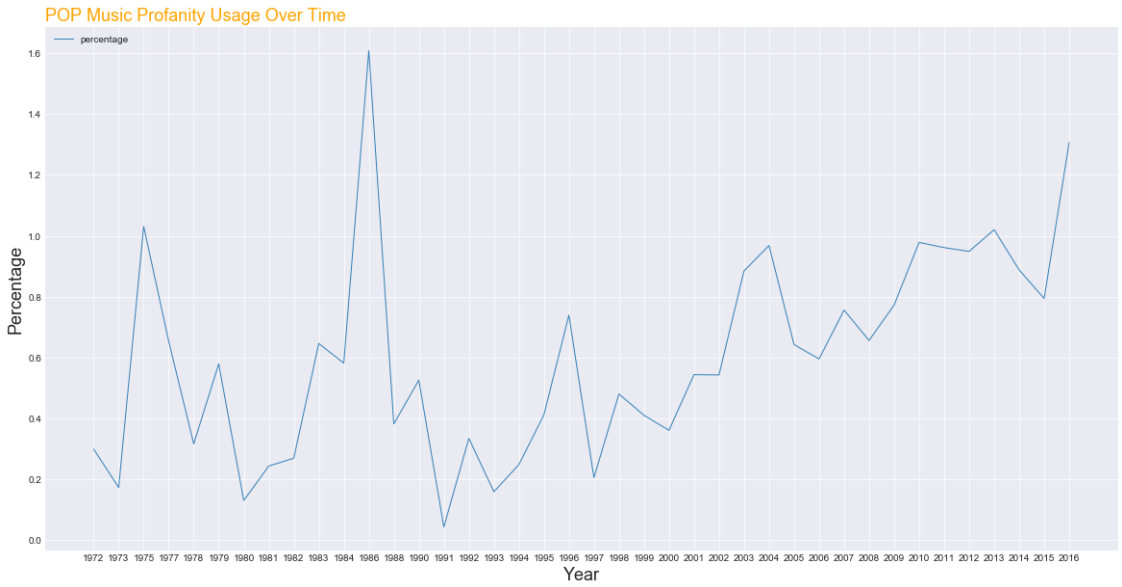
In [13]:
```
popByYear = getProfanityByYear('POP')
popByYear['percentage'] = (((popByYear['count']) / (popByYear['total_word_count'])) *
popByYear.head(200)
```

Out[13]:

	year	total_word_count	count	percentage
0	1972	1335	4	0.299625
1	1973	1152	2	0.173611
2	1975	582	6	1.030928
3	1977	5201	34	0.653720
4	1978	3786	12	0.316957

5	1979	2589	15	0.579374
6	1980	2291	3	0.130947
7	1981	6965	17	0.244078
8	1982	2225	6	0.269663
9	1983	1392	9	0.646552
10	1984	5842	34	0.581992
11	1986	2176	35	1.608456
12	1988	5493	21	0.382305
13	1990	9119	48	0.526374
14	1991	2276	1	0.043937
15	1992	2687	9	0.334946
16	1993	4384	7	0.159672
17	1994	12061	30	0.248736
18	1995	13103	54	0.412119
19	1996	7031	52	0.739582
20	1997	8734	18	0.206091
21	1998	13524	65	0.480627
22	1999	22639	93	0.410796
23	2000	11071	40	0.361304
24	2001	28855	157	0.544100
25	2002	46572	253	0.543245
26	2003	36173	320	0.884638
27	2004	62375	604	0.968337
28	2005	66708	429	0.643101
29	2006	1172706	6985	0.595631
30	2007	570342	4313	0.756213
31	2008	314568	2065	0.656456
32	2009	234168	1809	0.772522
33	2010	213616	2090	0.978391
34	2011	268444	2581	0.961467
35	2012	234263	2223	0.948933
36	2013	246793	2518	1.020288
37	2014	289682	2573	0.888215
38	2015	226140	1796	0.794198
39	2016	224373	2931	1.306307

In [14]:
```
from matplotlib.backends.backend_pdf import PdfPages
with PdfPages('InappropriatePOPLyricsByYear.pdf') as pdf_pages:
    plt.figure(figsize=(20, 10))
    plt.style.use('seaborn-darkgrid')
    plt.plot('year', 'percentage', data=popByYear,linewidth=1, alpha=0.9, label='perc
    plt.legend(loc=2, ncol=2)
    plt.title("POP Music Profanity Usage Over Time", loc='left', fontsize=18, fontweig
    plt.xlabel("Year", fontsize=18)
    plt.ylabel("Percentage", fontsize=18)
    pdf_pages.savefig()
    plt.show()
```



7 Profanity Keyword Over Time

Note Advisory

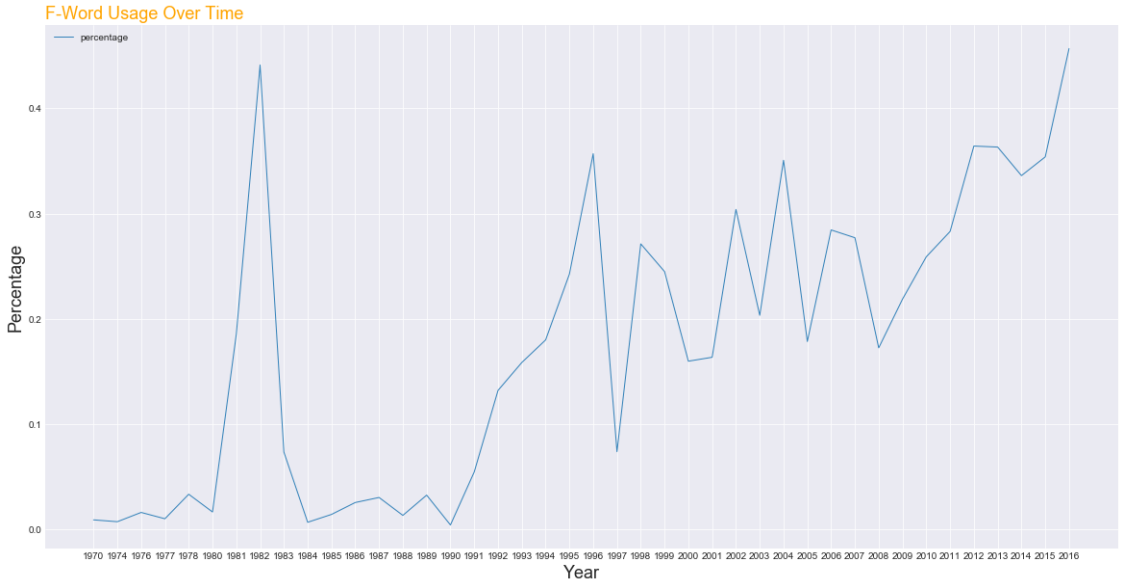
```
In [15]: fwordByYear = getWordByYear('FUCK')
         fwordByYear['percentage'] = (((fwordByYear['word_count']) / (fwordByYear['total_word_c
         fwordByYear.head(200)
```

Out[15]:

	year	word_count	total_word_count	percentage
0	1970	1	11290	0.008857
1	1974	1	14071	0.007107
2	1976	1	6281	0.015921
3	1977	2	20187	0.009907
4	1978	5	15061	0.033198
5	1980	3	18339	0.016359
6	1981	26	14033	0.185278
7	1982	75	16999	0.441202
8	1983	8	10925	0.073227
9	1984	1	15221	0.006570
10	1985	2	14263	0.014022
11	1986	4	15786	0.025339
12	1987	3	9955	0.030136
13	1988	2	15243	0.013121
14	1989	8	24723	0.032359
15	1990	4	100216	0.003991
16	1991	14	25740	0.054390
17	1992	68	51558	0.131890

18	1993	73	46084	0.158406
19	1994	101	56157	0.179853
20	1995	179	73843	0.242406
21	1996	249	69773	0.356872
22	1997	49	66549	0.073630
23	1998	204	75236	0.271147
24	1999	235	95998	0.244797
25	2000	187	117119	0.159667
26	2001	185	113222	0.163396
27	2002	509	167538	0.303812
28	2003	353	173683	0.203244
29	2004	1081	308311	0.350620
30	2005	874	490066	0.178343
31	2006	19657	6911620	0.284405
32	2007	15187	5482926	0.276987
33	2008	3073	1783379	0.172313
34	2009	2244	1026475	0.218612
35	2010	2740	1058391	0.258884
36	2011	2920	1031968	0.282955
37	2012	3912	1074282	0.364150
38	2013	3717	1023522	0.363158
39	2014	4129	1228703	0.336045
40	2015	3275	925607	0.353822
41	2016	4716	1032436	0.456784

In [16]:
```
from matplotlib.backends.backend_pdf import PdfPages
with PdfPages('F-WORD_BY_YEAR.pdf') as pdf_pages:
    plt.figure(figsize=(20, 10))
    plt.style.use('seaborn-darkgrid')
    plt.plot('year', 'percentage', data=fwordByYear,linewidth=1, alpha=0.9, label='pe
    plt.legend(loc=2, ncol=2)
    plt.title("F-Word Usage Over Time", loc='left', fontsize=18, fontweight=0, color=
    plt.xlabel("Year", fontsize=18)
    plt.ylabel("Percentage", fontsize=18)
    pdf_pages.savefig()
    plt.show()
```



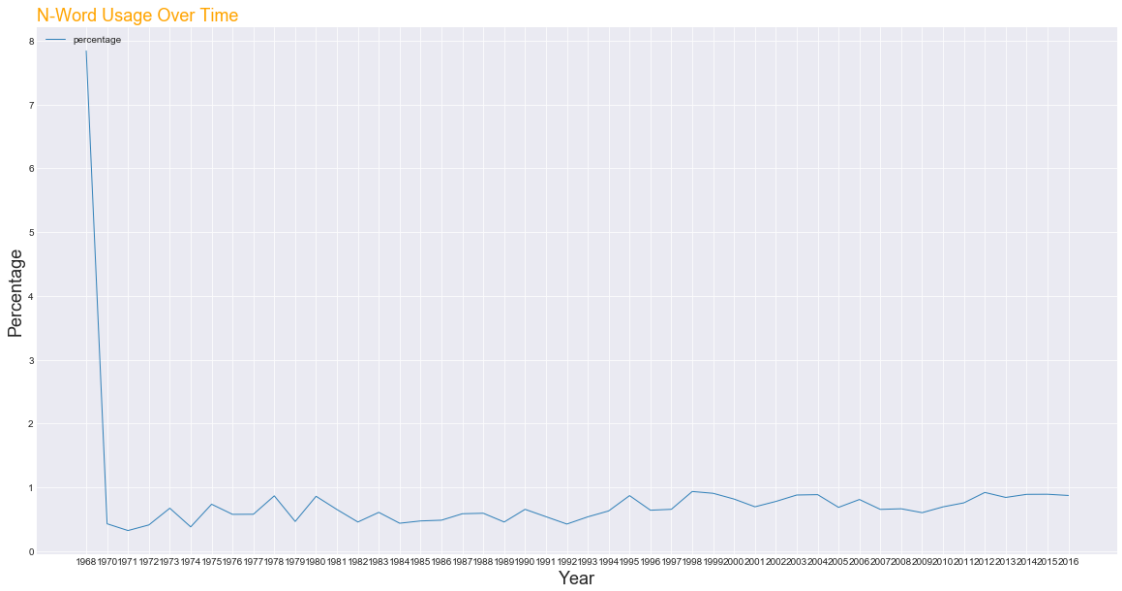

```
In [17]: nwordByYear = getWordByYear('NIG')
         nwordByYear['percentage'] = (((nwordByYear['word_count']) / (nwordByYear['total_word_c
         nwordByYear.head(200)
```

Out[17]:

	year	word_count	total_word_count	percentage
0	1968	4	51	7.843137
1	1970	48	11290	0.425155
2	1971	47	14737	0.318925
3	1972	60	14727	0.407415
4	1973	131	19576	0.669187
5	1974	53	14071	0.376661
6	1975	80	10932	0.731797
7	1976	36	6281	0.573157
8	1977	116	20187	0.574627
9	1978	130	15061	0.863156
10	1979	72	15609	0.461272
11	1980	157	18339	0.856099
12	1981	91	14033	0.648471
13	1982	77	16999	0.452968
14	1983	66	10925	0.604119
15	1984	66	15221	0.433611
16	1985	67	14263	0.469747
17	1986	76	15786	0.481439
18	1987	58	9955	0.582622
19	1988	90	15243	0.590435
20	1989	112	24723	0.453019
21	1990	653	100216	0.651593
22	1991	138	25740	0.536131

23	1992	217	51558	0.420885
24	1993	246	46084	0.533808
25	1994	352	56157	0.626814
26	1995	640	73843	0.866704
27	1996	445	69773	0.637783
28	1997	434	66549	0.652151
29	1998	701	75236	0.931735
30	1999	867	95998	0.903144
31	2000	952	117119	0.812848
32	2001	783	113222	0.691562
33	2002	1299	167538	0.775346
34	2003	1522	173683	0.876309
35	2004	2720	308311	0.882226
36	2005	3346	490066	0.682765
37	2006	55691	6911620	0.805759
38	2007	35679	5482926	0.650729
39	2008	11779	1783379	0.660488
40	2009	6151	1026475	0.599235
41	2010	7308	1058391	0.690482
42	2011	7777	1031968	0.753609
43	2012	9847	1074282	0.916612
44	2013	8579	1023522	0.838184
45	2014	10897	1228703	0.886870
46	2015	8222	925607	0.888282
47	2016	8974	1032436	0.869206

In [18]:
```
from matplotlib.backends.backend_pdf import PdfPages
with PdfPages('N-WORD_BY_YEAR.pdf') as pdf_pages:
    plt.figure(figsize=(20, 10))
    plt.style.use('seaborn-darkgrid')
    plt.plot('year', 'percentage', data=nwordByYear,linewidth=1, alpha=0.9, label='pe
    plt.legend(loc=2, ncol=2)
    plt.title("N-Word Usage Over Time", loc='left', fontsize=18, fontweight=0, color=
    plt.xlabel("Year", fontsize=18)
    plt.ylabel("Percentage", fontsize=18)
    pdf_pages.savefig()
    plt.show()
```




```
In [9]: conn.close()
```