In [90]:
```
from plotly import __version__
from plotly.offline import download_plotlyjs, init_notebook_mode, plot
, iplot
init_notebook_mode(connected=True)
import sys
import re
import vaderSentiment as vs
!{sys.executable} -m pip install textblob
from textblob import TextBlob
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
analyzer = SentimentIntensityAnalyzer()
```

```
Requirement already satisfied: textblob in ./anaconda3/lib/python3.6
/site-packages
Requirement already satisfied: nltk>=3.1 in ./anaconda3/lib/python3.
6/site-packages (from textblob)
Requirement already satisfied: six in ./anaconda3/lib/python3.6/site
-packages (from nltk>=3.1->textblob)
```

In [91]:
```
with open('1980.txt', 'r') as myfile:
    DietaryGuidelines1980=myfile.read()

with open('1985.txt', 'r') as myfile:
    DietaryGuidelines1985=myfile.read()

with open('1990.txt', 'r') as myfile:
    DietaryGuidelines1990=myfile.read()

with open('1995.txt', 'r') as myfile:
    DietaryGuidelines1995=myfile.read()

with open('2000.txt', 'r') as myfile:
    DietaryGuidelines2000=myfile.read()

with open('2005.txt', 'r') as myfile:
    DietaryGuidelines2005=myfile.read()

with open('2010.txt', 'r') as myfile:
    DietaryGuidelines2010=myfile.read()

with open('2015.txt', 'r') as myfile:
    DietaryGuidelines2015=myfile.read()
```

In [92]:
```
DietaryGuidelines1980=DietaryGuidelines1980.replace('\n_____
_\n','\n')
DietaryGuidelines1985=DietaryGuidelines1985.replace('\n_____
_\n','\n')
DietaryGuidelines1990=DietaryGuidelines1990.replace('\n_____
_\n','\n')
DietaryGuidelines1995=DietaryGuidelines1995.replace('\n_____
_\n','\n')
DietaryGuidelines2000=DietaryGuidelines2000.replace('\n_____
_\n','\n')
DietaryGuidelines2005=DietaryGuidelines2005.replace('\n_____
_\n','\n')
DietaryGuidelines2010=DietaryGuidelines2010.replace('\n_____
_\n','\n')
DietaryGuidelines2015=DietaryGuidelines2015.replace('\n_____
_\n','\n')
```

In [93]:
```
DietaryGuidelines1980 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
1980)
DietaryGuidelines1985 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
1985)
DietaryGuidelines1990 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
1990)
DietaryGuidelines1995 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
1995)
DietaryGuidelines2000 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
2000)
DietaryGuidelines2005 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
2005)
DietaryGuidelines2010 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
2010)
DietaryGuidelines2015 = re.sub(r'\n([a-z]+)',r' \1', DietaryGuidelines
2015)
```

In [94]:
```
DietaryGuidelines1980list1 = [sentence for sentence in DietaryGuidelin
es1980.split('\n')]

DietaryGuidelines1985list1 = [sentence for sentence in DietaryGuidelin
es1985.split('\n')]

DietaryGuidelines1990list1 = [sentence for sentence in DietaryGuidelin
es1990.split('\n')]

DietaryGuidelines1995list1 = [sentence for sentence in DietaryGuidelin
es1995.split('\n')]

DietaryGuidelines2000list1 = [sentence for sentence in DietaryGuidelin
es2000.split('\n')]

DietaryGuidelines2005list1 = [sentence for sentence in DietaryGuidelin
es2005.split('\n')]

DietaryGuidelines2010list1 = [sentence for sentence in DietaryGuidelin
es2010.split('\n')]

DietaryGuidelines2015list1 = [sentence for sentence in DietaryGuidelin
es2015.split('\n')]
```

In [95]:
```
DietaryGuidelines1980list2 = []
for i in DietaryGuidelines1980list1:
    DietaryGuidelines1980list2.append(i.split('.'))

    DietaryGuidelines1985list2 = []
for i in DietaryGuidelines1985list1:
    DietaryGuidelines1985list2.append(i.split('.'))

    DietaryGuidelines1990list2 = []
for i in DietaryGuidelines1990list1:
    DietaryGuidelines1990list2.append(i.split('.'))

    DietaryGuidelines1995list2 = []
for i in DietaryGuidelines1995list1:
    DietaryGuidelines1995list2.append(i.split('.'))

    DietaryGuidelines2000list2 = []
for i in DietaryGuidelines2000list1:
    DietaryGuidelines2000list2.append(i.split('.'))

    DietaryGuidelines2005list2 = []
for i in DietaryGuidelines2005list1:
    DietaryGuidelines2005list2.append(i.split('.'))

    DietaryGuidelines2010list2 = []
for i in DietaryGuidelines2010list1:
    DietaryGuidelines2010list2.append(i.split('.'))

    DietaryGuidelines2015list2 = []
for i in DietaryGuidelines2015list1:
    DietaryGuidelines2015list2.append(i.split('.'))
```

In [96]:
```
DietaryGuidelines1980FinalList = []
for i in DietaryGuidelines1980list2:
    for j in i:
        DietaryGuidelines1980FinalList.append(j)

DietaryGuidelines1985FinalList = []
for i in DietaryGuidelines1985list2:
    for j in i:
        DietaryGuidelines1985FinalList.append(j)


DietaryGuidelines1990FinalList = []
for i in DietaryGuidelines1990list2:
    for j in i:
        DietaryGuidelines1990FinalList.append(j)


DietaryGuidelines1995FinalList = []
for i in DietaryGuidelines1995list2:
    for j in i:
        DietaryGuidelines1995FinalList.append(j)


DietaryGuidelines2000FinalList = []
for i in DietaryGuidelines2000list2:
    for j in i:
        DietaryGuidelines2000FinalList.append(j)


DietaryGuidelines2005FinalList = []
for i in DietaryGuidelines2005list2:
    for j in i:
        DietaryGuidelines2005FinalList.append(j)


DietaryGuidelines2010FinalList = []
for i in DietaryGuidelines2010list2:
    for j in i:
        DietaryGuidelines2010FinalList.append(j)


DietaryGuidelines2015FinalList = []
for i in DietaryGuidelines2015list2:
    for j in i:
        DietaryGuidelines2015FinalList.append(j)
```

In [97]:
```
sugarlist1980 = []
for i in DietaryGuidelines1980FinalList:
    if 'sugar' in i:
        sugarlist1980.append(i)
#sugarlist1980

sugarlist1985 = []
for i in DietaryGuidelines1985FinalList:
    if 'sugar' in i:
        sugarlist1985.append(i)
#sugarlist1985

sugarlist1990 = []
for i in DietaryGuidelines1990FinalList:
    if 'sugar' in i:
        sugarlist1990.append(i)
#sugarlist1990

sugarlist1995 = []
for i in DietaryGuidelines1995FinalList:
    if 'sugar' in i:
        sugarlist1995.append(i)
#sugarlist1995

sugarlist2000 = []
for i in DietaryGuidelines2000FinalList:
    if 'sugar' in i:
        sugarlist2000.append(i)
#sugarlist2000

sugarlist2005 = []
for i in DietaryGuidelines2005FinalList:
    if 'sugar' in i:
        sugarlist2005.append(i)
#sugarlist2005

sugarlist2010 = []
for i in DietaryGuidelines2010FinalList:
    if 'sugar' in i:
        sugarlist2010.append(i)
#sugarlist2010

sugarlist2015 = []
for i in DietaryGuidelines2015FinalList:
    if 'sugar' in i:
        sugarlist2015.append(i)
#sugarlist2015
```

In [98]:
```
sugarsentimentlist1980 = []
for i in sugarlist1980:
        sugarsentimentlist1980.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist1980

sugarsentimentlist1985 = []
for i in sugarlist1985:
        sugarsentimentlist1985.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist1985

sugarsentimentlist1990 = []
for i in sugarlist1990:
        sugarsentimentlist1990.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist1990

sugarsentimentlist1995 = []
for i in sugarlist1995:
        sugarsentimentlist1995.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist1995

sugarsentimentlist2000 = []
for i in sugarlist2000:
        sugarsentimentlist2000.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist2000

sugarsentimentlist2005 = []
for i in sugarlist2005:
        sugarsentimentlist2005.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist2005

sugarsentimentlist2010 = []
for i in sugarlist2010:
        sugarsentimentlist2010.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist2010

sugarsentimentlist2015 = []
for i in sugarlist2015:
        sugarsentimentlist2015.append(TextBlob(i).sentiment.polarity)
#sugarsentimentlist2015
```

In [99]:
```
def mean(numbers):
    return float(sum(numbers)) / max(len(numbers), 1)

meansugarSentiment1980 = mean(sugarsentimentlist1980)

meansugarSentiment1985 = mean(sugarsentimentlist1985)

meansugarSentiment1990 = mean(sugarsentimentlist1990)

meansugarSentiment1995 = mean(sugarsentimentlist1995)

meansugarSentiment2000 = mean(sugarsentimentlist2000)

meansugarSentiment2005 = mean(sugarsentimentlist2005)

meansugarSentiment2010 = mean(sugarsentimentlist2010)

meansugarSentiment2015 = mean(sugarsentimentlist2015)


def normailizedMean(numbers):
    return float(float(sum(numbers)) / max(len(numbers), 1))/max(len(n
umbers), 1)
```

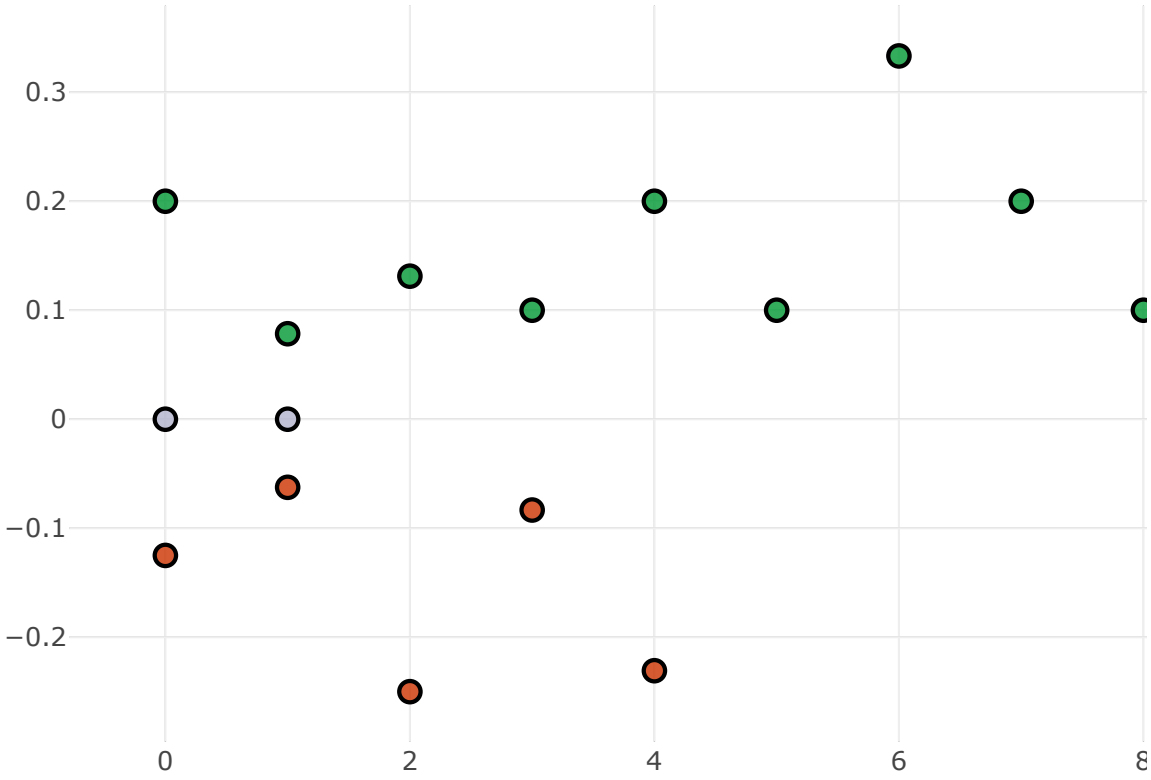
In [100]:
```
positivesugarsentimentlist1980 = []
negativesugarsentimentlist1980 = []
neutralsugarsentimentlist1980 = []
for i in sugarsentimentlist1980:
        if i>0:
                positivesugarsentimentlist1980.append(i)
        elif i==0:
                neutralsugarsentimentlist1980.append(i)
        else:
                negativesugarsentimentlist1980.append(i)

positivesugarsentimentlist1985 = []
negativesugarsentimentlist1985 = []
neutralsugarsentimentlist1985 = []
for i in sugarsentimentlist1985:
        if i>0:
                positivesugarsentimentlist1985.append(i)
        elif i==0:
                neutralsugarsentimentlist1985.append(i)
        else:
                negativesugarsentimentlist1985.append(i)

positivesugarsentimentlist1990 = []
negativesugarsentimentlist1990 = []
neutralsugarsentimentlist1990 = []
```

```
for i in sugarsentimentlist1990:
        if i>0:
                positivesugarsentimentlist1990.append(i)
        elif i==0:
                neutralsugarsentimentlist1990.append(i)
        else:
                negativesugarsentimentlist1990.append(i)

positivesugarsentimentlist1995 = []
negativesugarsentimentlist1995 = []
neutralsugarsentimentlist1995 = []
for i in sugarsentimentlist1995:
        if i>0:
                positivesugarsentimentlist1995.append(i)
        elif i==0:
                neutralsugarsentimentlist1995.append(i)
        else:
                negativesugarsentimentlist1995.append(i)

positivesugarsentimentlist2000 = []
negativesugarsentimentlist2000 = []
neutralsugarsentimentlist2000 = []
for i in sugarsentimentlist2000:
        if i>0:
                positivesugarsentimentlist2000.append(i)
        elif i==0:
                neutralsugarsentimentlist2000.append(i)
        else:
                negativesugarsentimentlist2000.append(i)

positivesugarsentimentlist2005 = []
negativesugarsentimentlist2005 = []
neutralsugarsentimentlist2005 = []
for i in sugarsentimentlist2005:
        if i>0:
                positivesugarsentimentlist2005.append(i)
        elif i==0:
                neutralsugarsentimentlist2005.append(i)
        else:
                negativesugarsentimentlist2005.append(i)

positivesugarsentimentlist2010 = []
negativesugarsentimentlist2010 = []
neutralsugarsentimentlist2010 = []
for i in sugarsentimentlist2010:
        if i>0:
                positivesugarsentimentlist2010.append(i)
        elif i==0:
                neutralsugarsentimentlist2010.append(i)
        else:
```

```
                negativesugarsentimentlist2010.append(i)

positivesugarsentimentlist2015 = []
negativesugarsentimentlist2015 = []
neutralsugarsentimentlist2015 = []
for i in sugarsentimentlist2015:
        if i>0:
                positivesugarsentimentlist2015.append(i)
        elif i==0:
                neutralsugarsentimentlist2015.append(i)
        else:
                negativesugarsentimentlist2015.append(i)
```

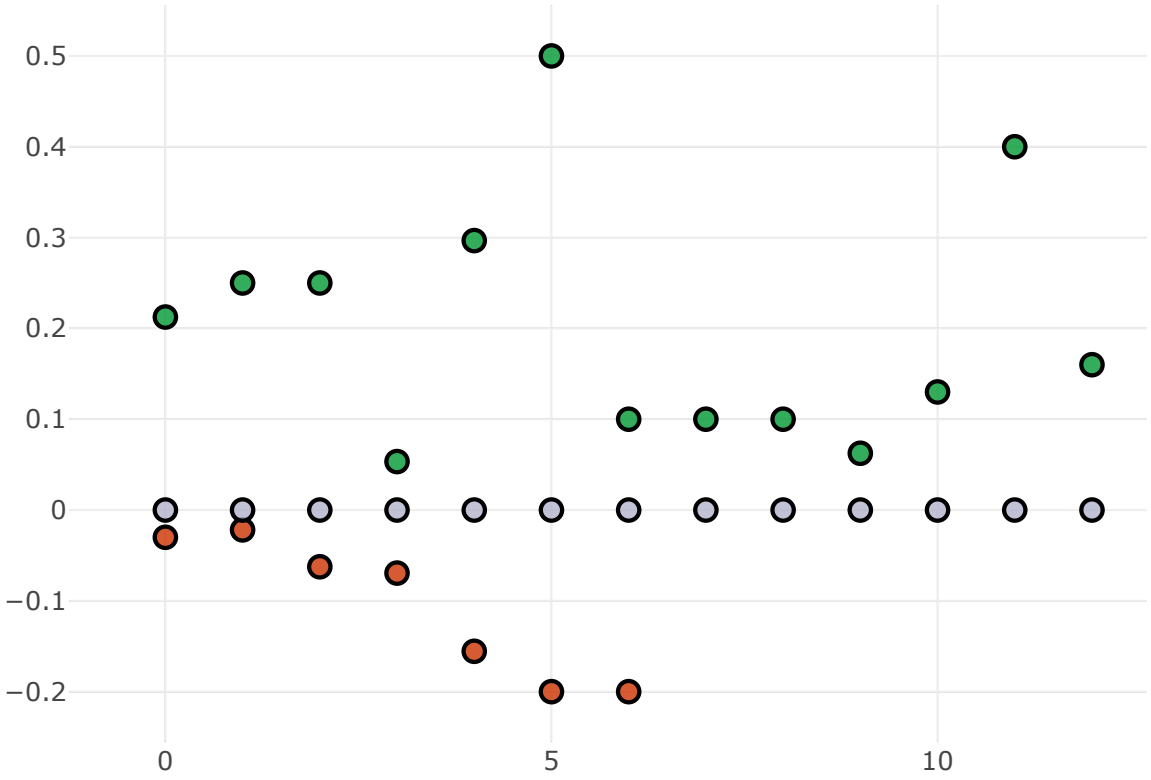
In [101]:
```
import plotly
plotly.tools.set_credentials_file(username='shivam.saith', api_key='xC
ijI2Ae8pZolWXkXMxN')
import plotly.plotly as py
import plotly.graph_objs as go

import numpy as np


trace0 = go.Scatter(
    x = np.arange(0,20),
    y = positivesugarsentimentlist1980,
    name = 'Positive',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(0, 153, 51, .8)',
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
    )
)

trace1 = go.Scatter(
    x = np.arange(0,20),
    y = negativesugarsentimentlist1980,
    name = 'Negative',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(204, 51, 0, .8)',
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
```

```
    )
)

trace2 = go.Scatter(
    x = np.arange(0,20),
    y = neutralsugarsentimentlist1980,
    name = 'Neutral',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(179, 179, 204, .8)',
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
    )
)
data = [trace0, trace1,trace2]
#data = [trace0]

layout = dict(title = 'Sugar sentiment scatter plot(1980)',
              yaxis = dict(zeroline = False),
              xaxis = dict(zeroline = False)
             )

fig = dict(data=data, layout=layout)
iplot(fig, filename='sugar-sentiment-scatter-1980')
```



In [102]:
```
import plotly
plotly.tools.set_credentials_file(username='shivam.saith', api_key='xC
ijI2Ae8pZolWXkXMxN')
import plotly.plotly as py
import plotly.graph_objs as go

import numpy as np


trace0 = go.Scatter(
    x = np.arange(0,100),
    y = positivesugarsentimentlist2000,
    name = 'Positive',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(0, 153, 51, .8)',
```

```
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
    )
)

trace1 = go.Scatter(
    x = np.arange(0,100),
    y = negativesugarsentimentlist2000,
    name = 'Negative',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(204, 51, 0, .8)',
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
    )
)

trace2 = go.Scatter(
    x = np.arange(0,100),
    y = neutralsugarsentimentlist2000,
    name = 'Neutral',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(179, 179, 204, .8)',
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
    )
)
data = [trace0, trace1,trace2]
#data = [trace0]

layout = dict(title = 'Sugar sentiment scatter plot(2015)',
              yaxis = dict(zeroline = False),
              xaxis = dict(zeroline = False)
             )

fig = dict(data=data, layout=layout)
iplot(fig, filename='sugar-sentiment-scatter-2015')
```



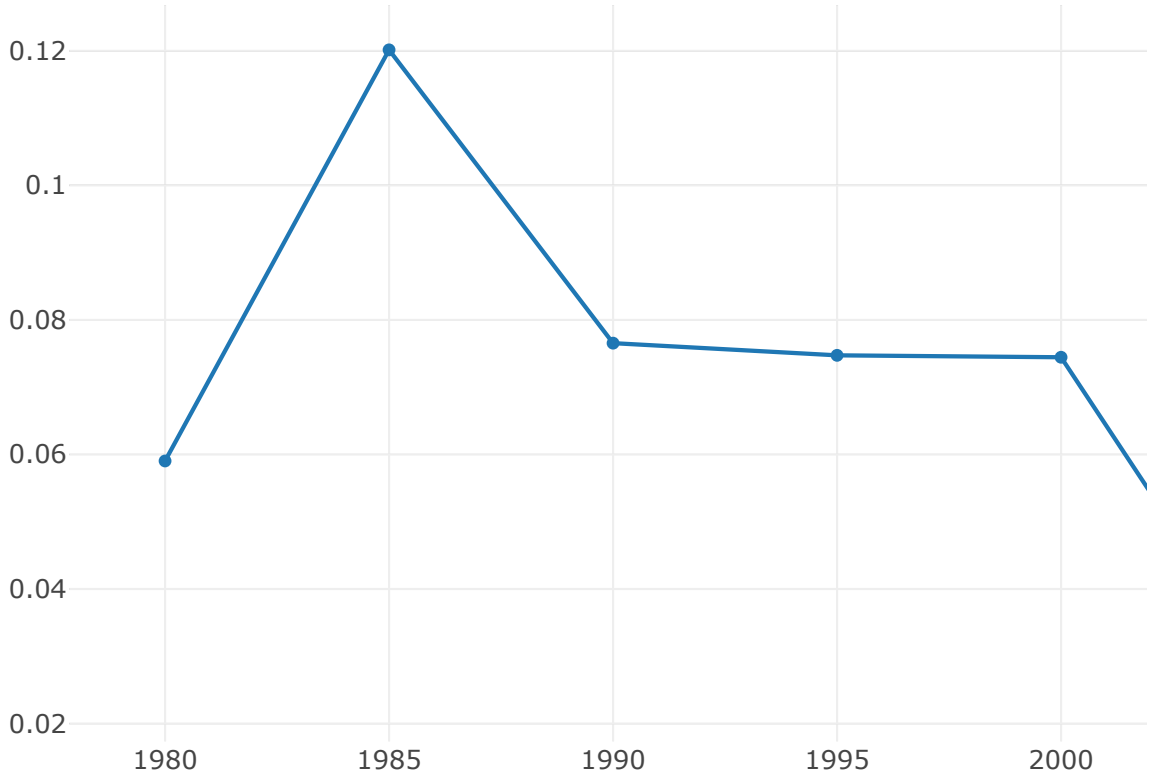
In [103]:
```
len(positivesugarsentimentlist2015)
```

Out[103]: 60

In [104]:
```
sugarSentimentAcrossYears = []

sugarSentimentAcrossYears.append(meansugarSentiment1980)
sugarSentimentAcrossYears.append(meansugarSentiment1985)
sugarSentimentAcrossYears.append(meansugarSentiment1990)
sugarSentimentAcrossYears.append(meansugarSentiment1995)
sugarSentimentAcrossYears.append(meansugarSentiment2000)
sugarSentimentAcrossYears.append(meansugarSentiment2005)
sugarSentimentAcrossYears.append(meansugarSentiment2010)
sugarSentimentAcrossYears.append(meansugarSentiment2015)


print(meansugarSentiment1980)

print(meansugarSentiment1985)

print(meansugarSentiment1990)

print(meansugarSentiment1995)

print(meansugarSentiment2000)

print(meansugarSentiment2005)

print(meansugarSentiment2010)

print(meansugarSentiment2015)
```

```
0.05903475274725276
0.12014715034069876
0.07654761904761904
0.07473804023804025
0.07445263532763534
0.024055859264192597
0.03384568632250752
0.07404909967809967
```

In [105]:
```
import plotly.plotly as py
import plotly.graph_objs as go

import pandas as pd
import numpy as np

df = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/ma
ster/finance-charts-apple.csv")

data = [go.Scatter(
          x=np.arange(1980,2020,5),
          y=sugarSentimentAcrossYears)]

iplot(data)
```



In [106]:
```
def getNutrientRelevantStatementsPolarityListUsingTextBlob(year, nutri
entName):
    with open(year + '.txt', 'r') as myfile:
        text = myfile.read()
    text1 =text.replace('\n_____\n','\n')
    text2 = re.sub(r'\n([a-z]+)',r' \1', text1)
    list1 = [sentence for sentence in text2.split('\n')]
    list2 = []
    for i in list1:
        list2.append(i.split('.'))

    finalList = []

    for i in list2:
            for j in i:
                finalList.append(j)

    nutrientStatementList = []

    for i in finalList:
        if nutrientName.lower() in i:
            nutrientStatementList.append(i)


    statementsPolarityList = []

    for i in nutrientStatementList:
            statementsPolarityList.append(TextBlob(i).sentiment.polari
ty)


    return statementsPolarityList;


#with open('1980thin.pdf.txt', 'r') as myfile:
 #   DietaryGuidelines1980=myfile.read()

test = getNutrientRelevantStatementsPolarityListUsingTextBlob(year = '
2000',nutrientName = 'vitamin' )
print(test)
```

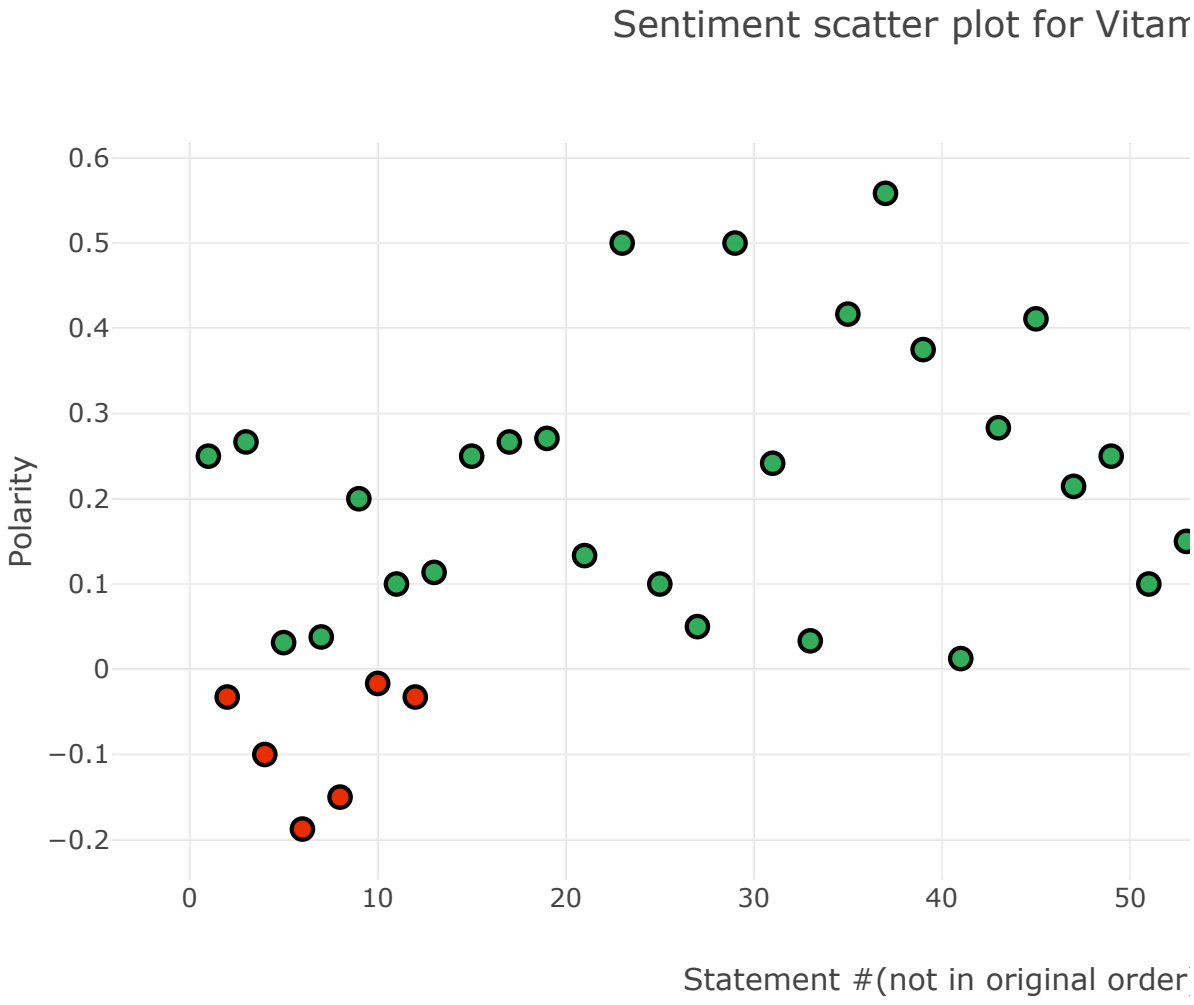
```
[0.0, 0.0, -0.01333333333333333, 0.19, 0.1875, 0.3333333333333333, 0
.125, 0.15625, 0.0, -0.010416666666666671, -0.012500000000000011, 0.
06666666666666665, 0.0, 0.10714285714285714, 0.19375, -0.2875, 0.325
, -0.11666666666666665, 0.24375, 0.6875, 1.0, 0.0, 0.0, 0.5, -0.1, -
0.2, 0.16]
```

In [107]:
```
def getPositivePolarityList(nutrientPolarityList):
    positivePolarityList = []
    for i in nutrientPolarityList:
        if i>0:
                positivePolarityList.append(i)
    return positivePolarityList;

def getNegativePolarityList(nutrientPolarityList):
    negativePolarityList = []
    for i in nutrientPolarityList:
        if i<0:
                negativePolarityList.append(i)
    return negativePolarityList;

def getNeutralPolarityList(nutrientPolarityList):
    neutralPolarityList = []
    for i in nutrientPolarityList:
        if i==0:
                neutralPolarityList.append(i)
    return neutralPolarityList;
```

In [108]:
```
def plotPolarityScatterPlot (nutrient,nutrientStatementPolarityList,ye
ar):
    import plotly
    plotly.tools.set_credentials_file(username='shivam.saith', api_key
='xCijI2Ae8pZolWXkXMxN')
    import plotly.plotly as py
    import plotly.graph_objs as go

    import numpy as np


    trace0 = go.Scatter(
        x = np.arange(1,1000,2),
        y = getPositivePolarityList(nutrientStatementPolarityList),
        name = 'Positive',
        mode = 'markers',
        marker = dict(
            size = 10,
            color = 'rgba(0, 153, 51, .8)',
            line = dict(
                width = 2,
                color = 'rgb(0, 0, 0)'
            )
        )
    )

    trace1 = go.Scatter(
```

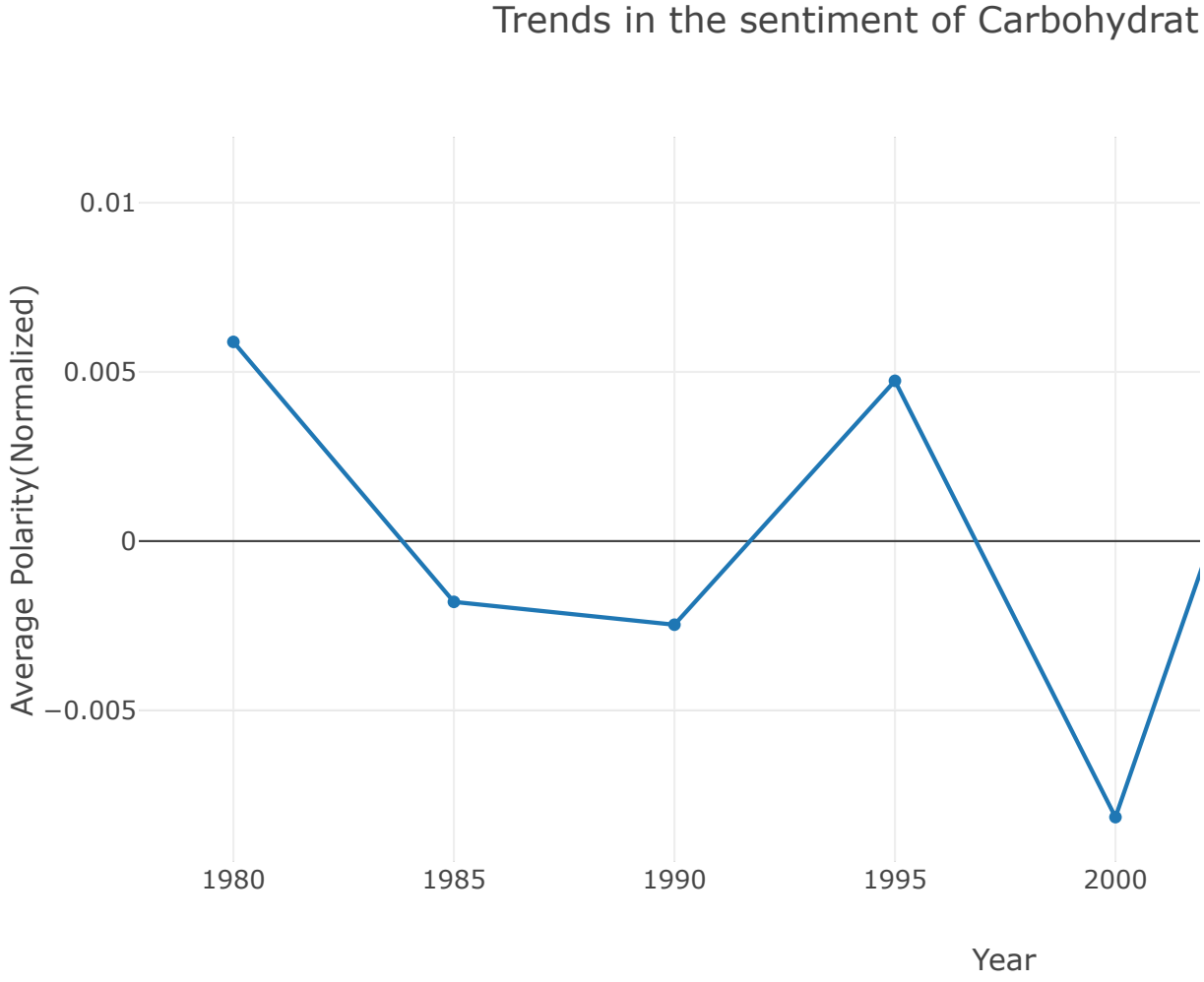
```
        x = np.arange(2,1001,2),
        y = getNegativePolarityList(nutrientStatementPolarityList),
        name = 'Negative',
        mode = 'markers',
        marker = dict(
            size = 10,
            color = 'rgb(230, 46, 0)',
            line = dict(
                width = 2,
                color = 'rgb(0, 0, 0)'
            )
        )
    )

    trace2 = go.Scatter(
        x = np.arange(3,1002,3),
        y = getNeutralPolarityList(nutrientStatementPolarityList),
        name = 'Neutral',
        mode = 'markers',
        marker = dict(
            size = 10,
            color = 'rgba(179, 179, 204, .8)',
            line = dict(
                width = 2,
                color = 'rgb(0, 0, 0)'
            )
        )
    )
    data = [trace0, trace1]
    #data = [trace0]

    layout = dict(title = 'Sentiment scatter plot for ' + nutrient + '
' + '(' + year +')',
                  yaxis = dict(title = 'Polarity',zeroline = False),
                  xaxis = dict(title = 'Statement #(not in original or
der)',zeroline = False)
                 )

    fig = dict(data=data, layout=layout)
    return fig;
```

In [109]:
```
requiredNutrient = 'Vitamin'
requiredYear = '2010'

iplot(plotPolarityScatterPlot(nutrient = requiredNutrient,nutrientStat
ementPolarityList = getNutrientRelevantStatementsPolarityListUsingText
Blob(year = requiredYear,nutrientName = requiredNutrient ), year = req
uiredYear), filename='sugar-polarity-scatter-2015')
```



In [110]:
```
def getNutrientPolarityTrendLineUsingTextBlob (nutrient):
    import numpy as np
    yearsList = np.arange(1980,2020,5)
    polarityMeansAcrossYearsList = []
    for i in yearsList:
        polarityMeansAcrossYearsList.append(normailizedMean(getNutrien
tRelevantStatementsPolarityListUsingTextBlob(year=str(i),nutrientName
= nutrient)))
    data = [go.Scatter(
          x=np.arange(1980,2020,5),
          y=polarityMeansAcrossYearsList)]
    layout = dict(
         title = "Trends in the sentiment of " + nutrient + '(s) ' + '
(1980-2015)' ,
         yaxis = dict(title = 'Average Polarity(Normalized)'),
         xaxis = dict(title = 'Year')
    )

    fig = dict(data=data, layout=layout)
    return fig;
```

In [111]:
```
import plotly.plotly as py
init_notebook_mode(connected=True)

nutrientName = 'Carbohydrate'

iplot(getNutrientPolarityTrendLineUsingTextBlob(nutrientName))
```


Trends in the sentiment of Carbohydrat
Average Polarity(Normalized)
0.01
0.005
0
−0.005
1980
1985
1990
1995
2000
Year

In [125]:
```
def getNutrientRelevantStatementsPolarityDictionaryUsingTextBlob(year,
nutrientName):
    with open(year + '.txt', 'r') as myfile:
        text = myfile.read()
    text1 =text.replace('\n_____\n','\n')
    text2 = re.sub(r'\n([a-z]+)',r' \1', text1)
    list1 = [sentence for sentence in text2.split('\n')]
    list2 = []
    for i in list1:
        list2.append(i.split('.'))



    finalList = []

    for i in list2:
            for j in i:
                finalList.append(j)

    nutrientStatementList = []

    for i in finalList:
        if nutrientName.lower() in i:
            nutrientStatementList.append(i)


    dict = {}

    for i in nutrientStatementList:
            dict[i] = TextBlob(i).sentiment.polarity


    return dict;
```

In [127]:
```
def getNutrientRelevantStatementsPolarityDictionaryUsingVader(year, nu
trientName):
    with open(year + '.txt', 'r') as myfile:
        text = myfile.read()
    text1 =text.replace('\n_____\n','\n')
    text2 = re.sub(r'\n([a-z]+)',r' \1', text1)
    list1 = [sentence for sentence in text2.split('\n')]
    list2 = []
    for i in list1:
        list2.append(i.split('.'))



    finalList = []

    for i in list2:
            for j in i:
                finalList.append(j)

    nutrientStatementList = []

    for i in finalList:
        if nutrientName.lower() in i:
            nutrientStatementList.append(i)


    dict = {}

    for i in nutrientStatementList:
            dict[i] = (analyzer.polarity_scores(i))['compound']


    return dict;
```

In [130]:
```
for statement, polarity in getNutrientRelevantStatementsPolarityDictio
naryUsingTextBlob(year = '2015', nutrientName = 'Carbohydrate').items(
):
    print('{} ==> {}'.format(statement, polarity))
```

NOTE: The total eating pattern should not exceed Dietary Guidelines limits for intake of calories from added sugars and saturated fats a nd alcohol and should be within the Acceptable Macronutrient Distrib ution Ranges for calories from protein, carbohydrate, and total fats ==> 0.0
On average, carbohydrates and protein contain 4 calories per gram, fats contain 9 calories per gram, and alcohol has 7 calories per gra m ==> -0.15
Strong evidence from mostly prospective cohort studies but also ran domized controlled trials has shown that eating patterns that patter ns can accommodate other nutrient- dense foods with small amounts of added sugars, such as whole-grain breakfast cereals or fat-free yogu rt, as long as calories from added sugars do not exceed 10 percent p er day, total carbohydrate intake remains within the AMDR, and total calorie intake remains within limits ==> 0.06354166666666666
with carbohydrates reduces blood levels ==> 0.0
Dietary Guidelines to limit consumption saturated fats with carbohyd rates is not of dietary cholesterol to 300 mg per day associated wit h reduced risk of CVD ==> 0.0
is not included in the 2015 edition, but Additional research is nee ded to determine this change does not suggest that dietary whether t his relationship is consistent cholesterol is no longer important to consider across categories of carbohydrates (e ==> 0.325
The OmniHeart Trial found that replacing some of the carbohydrates in DASH with the same amount of either protein or unsaturated fats l owered blood pressure and LDL-cholesterol levels more than the origi nal DASH dietary pattern ==> 0.2916666666666667
Additionally, healthy eating patterns can be flexible with respect to the intake of carbohydrate, protein, and fat within the context o f the AMDR ==> 0.5
Alcohol for additional guidance); and calories from protein, carboh ydrate, and total fats should be within the Acceptable Macronutrient Distribution Ranges (AMDRs) ==> 0.0
Dietary fiber consists of nondigestible carbohydrates and lignin th at are intrinsic and intact in plants (i ==> 0.0
Functional fiber consists of isolated, nondigestible carbohydrates that have beneficial physiological effects in humans ==> 0.0
• Estimated Average Requirements (EAR)—The average daily Diabetes—A disorder of metabolism— nutrient intake level estimated to the way t he body uses digested food meet the requirement of half the (specifi cally carbohydrate) for growth and healthy individuals in a particul ar energy ==> 0.03333333333333333
, carbohydrate, ==> 0.0
protein, fats, carbohydrates, and alcohol ==> 0.0

In [131]:
```
for statement, polarity in getNutrientRelevantStatementsPolarityDictio
naryUsingVader(year = '2015', nutrientName = 'Carbohydrate').items():
    print('{} ==> {}'.format(statement, polarity))
```

```
NOTE: The total eating pattern should not exceed Dietary Guidelines
limits for intake of calories from added sugars and saturated fats a
nd alcohol and should be within the Acceptable Macronutrient Distrib
ution Ranges for calories from protein, carbohydrate, and total fats
==> 0.3182
 On average, carbohydrates and protein contain 4 calories per gram,
fats contain 9 calories per gram, and alcohol has 7 calories per gra
m ==> 0.0
 Strong evidence from mostly prospective cohort studies but also ran
domized controlled trials has shown that eating patterns that patter
ns can accommodate other nutrient- dense foods with small amounts of
added sugars, such as whole-grain breakfast cereals or fat-free yogu
rt, as long as calories from added sugars do not exceed 10 percent p
er day, total carbohydrate intake remains within the AMDR, and total
calorie intake remains within limits ==> 0.2846
 with carbohydrates reduces blood levels ==> 0.0
Dietary Guidelines to limit consumption saturated fats with carbohyd
rates is not of dietary cholesterol to 300 mg per day associated wit
h reduced risk of CVD ==> -0.2732
 is not included in the 2015 edition, but Additional research is nee
ded to determine this change does not suggest that dietary whether t
his relationship is consistent cholesterol is no longer important to
consider across categories of carbohydrates (e ==> -0.1531
 The OmniHeart Trial found that replacing some of the carbohydrates
in DASH with the same amount of either protein or unsaturated fats l
owered blood pressure and LDL-cholesterol levels more than the origi
nal DASH dietary pattern ==> -0.0352
 Additionally, healthy eating patterns can be flexible with respect
to the intake of carbohydrate, protein, and fat within the context o
f the AMDR ==> 0.7717
 Alcohol for additional guidance); and calories from protein, carboh
ydrate, and total fats should be within the Acceptable Macronutrient
Distribution Ranges (AMDRs) ==> 0.3182
 Dietary fiber consists of nondigestible carbohydrates and lignin th
at are intrinsic and intact in plants (i ==> 0.2023
 Functional fiber consists of isolated, nondigestible carbohydrates
that have beneficial physiological effects in humans ==> 0.1531
• Estimated Average Requirements (EAR)—The average daily Diabetes—A
disorder of metabolism— nutrient intake level estimated to the way t
he body uses digested food meet the requirement of half the (specifi
cally carbohydrate) for growth and healthy individuals in a particul
ar energy ==> 0.5719
, carbohydrate, ==> 0.0
 protein, fats, carbohydrates, and alcohol ==> 0.0
```

In [114]:
```
from textblob import TextBlob
from textblob.sentiments import NaiveBayesAnalyzer
opinion = TextBlob("During digestion all carbohydrates except fiber br
eak down into sugars", analyzer=NaiveBayesAnalyzer())
opinion.sentiment
```

Out[114]:
```
Sentiment(classification='neg', p_pos=0.06619867710721464, p_neg=0.9
338013228927862)
```

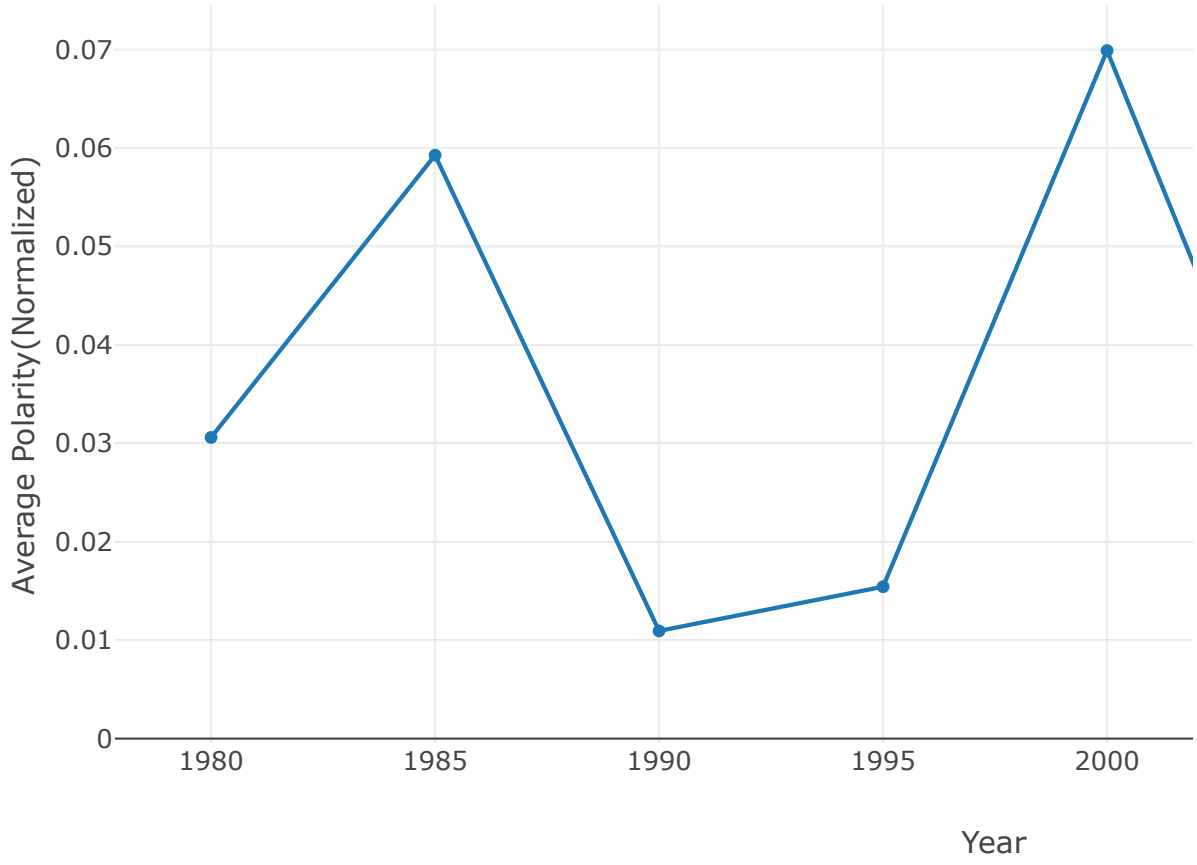
In [115]:
```
for statement, polarity in getNutrientRelevantStatementsPolarityDictio
naryUsingTextBlob(year = '2000', nutrientName = 'Carbohydrate').items(
):
    print('{} ==> {}'.format(statement, polarity))
```

```
The carbohydrates, fats, and proteins in food supply energy, which i
s measured in calories ==> 0.0
 oats) They provide vitamins, minerals, carbohydrates (starch and di
etary fiber), and other substances that are important for good healt
h ==> 0.325
 are carbohydrates Dietary carbohydrates and a source also include o
f energy the complex carbohydrates starch and dietary fiber ==> -0.3
 During digestion all carbohydrates except fiber break down into sug
ars ==> -0.15555555555555559
```

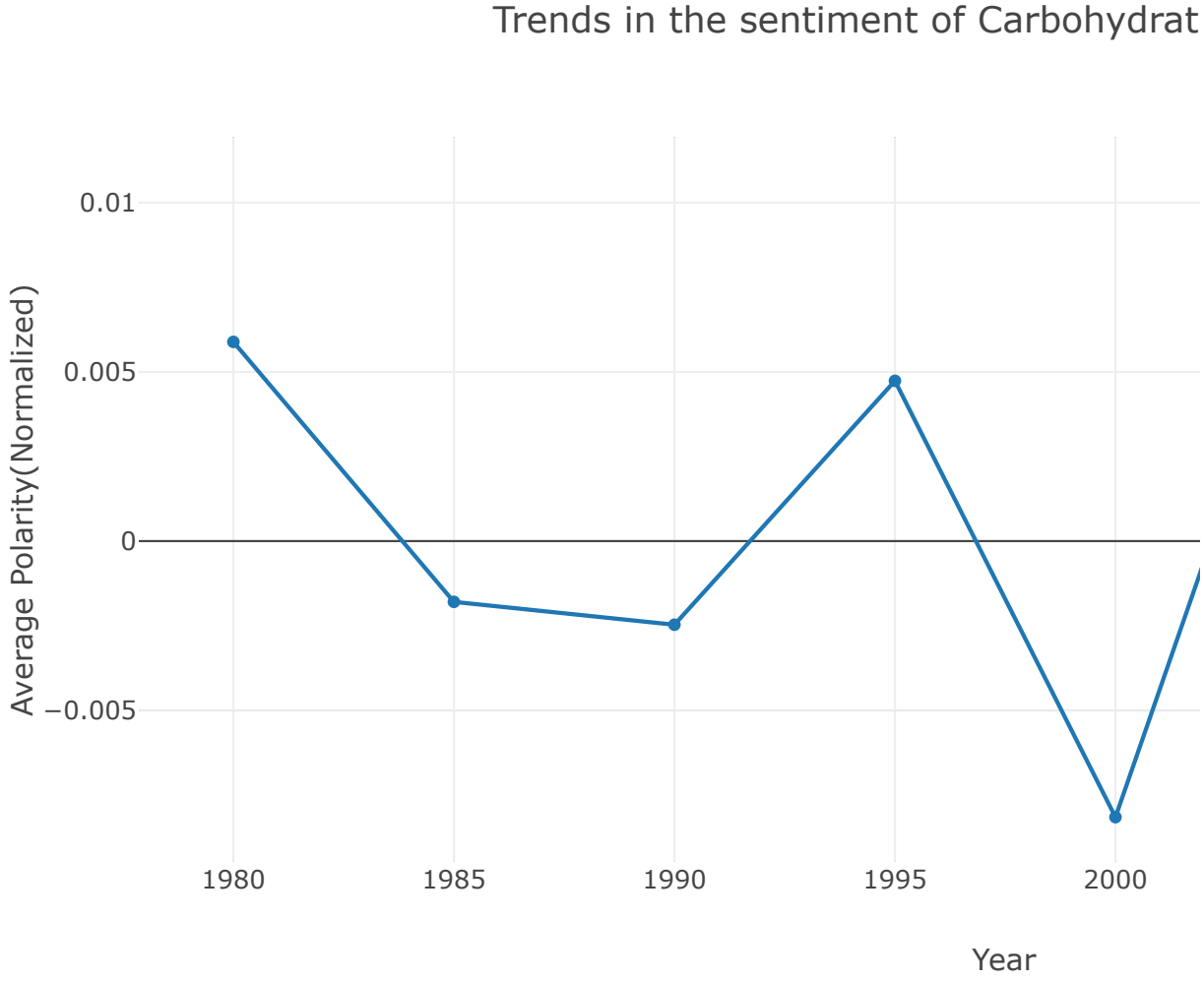
In [116]:
```
def getNutrientRelevantStatementsPolarityListUsingVader(year, nutrient
Name):
    with open(year + '.txt', 'r') as myfile:
        text = myfile.read()
    text1 =text.replace('\n_____\n','\n')
    text2 = re.sub(r'\n([a-z]+)',r' \1', text1)
    list1 = [sentence for sentence in text2.split('\n')]
    list2 = []
    for i in list1:
        list2.append(i.split('.'))

    finalList = []

    for i in list2:
            for j in i:
                finalList.append(j)

    nutrientStatementList = []

    for i in finalList:
        if nutrientName.lower() in i:
            nutrientStatementList.append(i)


    statementsPolarityList = []

    for i in nutrientStatementList:
            statementsPolarityList.append((analyzer.polarity_scores(i)
)['compound'])


    return statementsPolarityList;
```

In [117]:
```
def getNutrientPolarityTrendLineUsingVader (nutrient):
    import numpy as np
    yearsList = np.arange(1980,2020,5)
    polarityMeansAcrossYearsList = []
    for i in yearsList:
        polarityMeansAcrossYearsList.append(normailizedMean(getNutrien
tRelevantStatementsPolarityListUsingVader(year=str(i),nutrientName = n
utrient)))
    data = [go.Scatter(
          x=np.arange(1980,2020,5),
          y=polarityMeansAcrossYearsList)]
    layout = dict(
         title = "Trends in the sentiment of " + nutrient + '(s) ' + '
(1980-2015)' ,
         yaxis = dict(title = 'Average Polarity(Normalized)'),
         xaxis = dict(title = 'Year')
    )

    fig = dict(data=data, layout=layout)
    return fig;
```

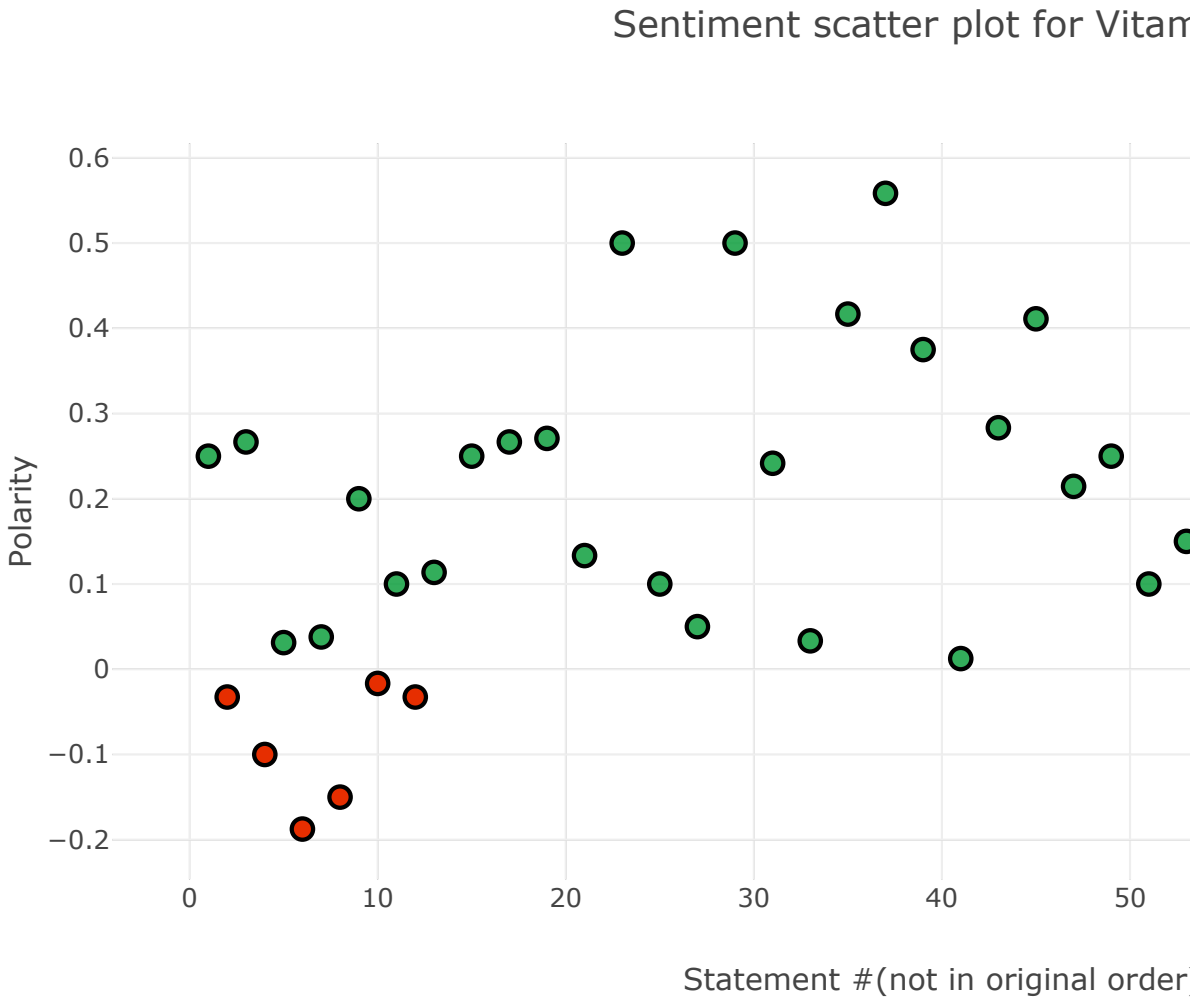
In [118]:
```
import plotly.plotly as py
init_notebook_mode(connected=True)

nutrientName = 'Carbohydrate'

iplot(getNutrientPolarityTrendLineUsingVader(nutrientName))
```



In [119]:
```
import plotly.plotly as py
init_notebook_mode(connected=True)

nutrientName = 'Carbohydrate'

iplot(getNutrientPolarityTrendLineUsingTextBlob(nutrientName))
```


Trends in the sentiment of Carbohydrat
Average Polarity(Normalized)
0.01
0.005
0
−0.005
1980
1985
1990
1995
2000
Year

In [120]:
```
requiredNutrient = 'Vitamin'
requiredYear = '2010'

iplot(plotPolarityScatterPlot(nutrient = requiredNutrient,nutrientStat
ementPolarityList = getNutrientRelevantStatementsPolarityListUsingText
Blob(year = requiredYear,nutrientName = requiredNutrient ), year = req
uiredYear), filename='sugar-polarity-scatter-2015')
```



In [121]:
```
requiredNutrient = 'Vitamin'
requiredYear = '2010'

iplot(plotPolarityScatterPlot(nutrient = requiredNutrient,nutrientStat
ementPolarityList = getNutrientRelevantStatementsPolarityListUsingVade
r(year = requiredYear,nutrientName = requiredNutrient ), year = requir
edYear), filename='sugar-polarity-scatter-2015')
```