In [1]:

```
%matplotlib inline

import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer
```

In [2]:

```
import json
import pandas as pd
final = pd.read_json('reviews_Movies_and_TV_5.json.gz', lines=True)
```

In [3]:
```
1 final
```

Out[3]:

	reviewerID	asin	reviewerName	helpful	reviewText	overall	summary	unixReviewTime	reviewTime
0	ADZPIG9QOCDG5	0005019281	Alice L. Larson "alice-loves-books"	[0, 0]	This is a charming version of the classic Dick...	4	good version of a classic	1203984000	02 26, 2008
1	A35947ZP82G7JH	0005019281	Amarah Strack	[0, 0]	It was good but not as emotionally moving as t...	3	Good but not as moving	1388361600	12 30, 2013
2	A3UORV8A9D5L2E	0005019281	Amazon Customer	[0, 0]	Don't get me wrong, Winkler is a wonderful cha...	3	Winkler's Performance was ok at best!	1388361600	12 30, 2013
3	A1VKW06X1O2X7V	0005019281	Amazon Customer "Softmill"	[0, 0]	Henry Winkler is very good in this twist on th...	5	It's an enjoyable twist on the classic story	1202860800	02 13, 2008
4	A3R27T4HADWFFJ	0005019281	BABE	[0, 0]	This is one of the best Scrooge movies out. H...	4	Best Scrooge yet	1387670400	12 22, 2013
...	...	...	...	...	...	...	...	...	...
1697528	AV657BUYHHXZ2	B00LT1JHLW	Mike Rules "Mike"	[1, 14]	wow $269.99 for the entire series on Blu Ray??...	1	Way to Expensive!! WB = GREED	1406073600	07 23, 2014
1697529	A17W587EH23J0Q	B00LT1JHLW	Ron2900 "Ron"	[32, 48]	Finally, the holy grail of tv-on-dvd boxsets i...	5	HOLY BAT-BOXSET, BATMAN... I never thought thi...	1405641600	07 18, 2014
1697530	A3DE438TF1A958	B00LT1JHLW	thomas henry	[3, 10]	Could this be a true or I'm i dreaming batman ...	5	prayers have been answered because batman 60s ...	1405728000	07 19, 2014
1697531	A2RWCXDMANY0LW	B00LT1JHLW	wheev	[0, 4]	I've been a fan of the series since I was a yo...	5	can't Wait!	1405987200	07 22, 2014
1697532	A3ROPC55BE2OM9	B00LT1JHLW	WingLT	[11, 23]	People seriously need to wake up and realize t...	5	The Price is Insane? People Really Need to Wak...	1405728000	07 19, 2014

1697533 rows × 9 columns

In [4]:

```
# find sentences containing HTML tags
import re
i=0;
for sent in final['reviewText'].values:
    if (len(re.findall('<.*?>', sent))):
        print(i)
        print(sent)
        break;
    i += 1;
```

In [ ]:
```
1 final
```

Out[5]:

	reviewerID	asin	reviewerName	helpful	reviewText	overall	summary	unixReviewTime	reviewTime
0	ADZPIG9QOCDG5	0005019281	Alice L. Larson "alice-loves-books"	[0, 0]	This is a charming version of the classic Dick...	4	good version of a classic	1203984000	02 26, 2008
1	A35947ZP82G7JH	0005019281	Amarah Strack	[0, 0]	It was good but not as emotionally moving as t...	3	Good but not as moving	1388361600	12 30, 2013
2	A3UORV8A9D5L2E	0005019281	Amazon Customer	[0, 0]	Don't get me wrong, Winkler is a wonderful cha...	3	Winkler's Performance was ok at best!	1388361600	12 30, 2013
3	A1VKW06X1O2X7V	0005019281	Amazon Customer "Softmill"	[0, 0]	Henry Winkler is very good in this twist on th...	5	It's an enjoyable twist on the classic story	1202860800	02 13, 2008
4	A3R27T4HADWFFJ	0005019281	BABE	[0, 0]	This is one of the best Scrooge movies out. H...	4	Best Scrooge yet	1387670400	12 22, 2013
...	...	...	...	...	...	...	...	...	...
1697528	AV657BUYHHXZ2	B00LT1JHLW	Mike Rules "Mike"	[1, 14]	wow $269.99 for the entire series on Blu Ray??...	1	Way to Expensive!! WB = GREED	1406073600	07 23, 2014
1697529	A17W587EH23J0Q	B00LT1JHLW	Ron2900 "Ron"	[32, 48]	Finally, the holy grail of tv-on-dvd boxsets i...	5	HOLY BAT-BOXSET, BATMAN... I never thought thi...	1405641600	07 18, 2014
1697530	A3DE438TF1A958	B00LT1JHLW	thomas henry	[3, 10]	Could this be a true or I'm i dreaming batman ...	5	prayers have been answered because batman 60s ...	1405728000	07 19, 2014
1697531	A2RWCXDMANY0LW	B00LT1JHLW	wheev	[0, 4]	I've been a fan of the series since I was a yo...	5	can't Wait!	1405987200	07 22, 2014
1697532	A3ROPC55BE2OM9	B00LT1JHLW	WingLT	[11, 23]	People seriously need to wake up and realize t...	5	The Price is Insane? People Really Need to Wak...	1405728000	07 19, 2014

1697533 rows × 9 columns

In [ ]:
```
df = final[~final['reviewText'].isnull()]
```

In [ ]:
```
from textblob import TextBlob
def preprocess(ReviewText):
    ReviewText = ReviewText.str.replace("(<br/>)", "")
    ReviewText = ReviewText.str.replace('(<a).*(>).*(</a>)', '')
    ReviewText = ReviewText.str.replace('(&amp)', '')
    ReviewText = ReviewText.str.replace('(&gt)', '')
    ReviewText = ReviewText.str.replace('(&lt)', '')
    ReviewText = ReviewText.str.replace('(\xa0)', ' ')
    return ReviewText
df['reviewText'] = preprocess(df['reviewText'])

df['polarity'] = df['reviewText'].map(lambda text: TextBlob(text).sentiment.polarity)
df['review_len'] = df['reviewText'].astype(str).apply(len)
df['word_count'] = df['reviewText'].apply(lambda x: len(str(x).split()))
```

In [ ]:
```
print('5 random reviews with the highest positive sentiment polarity: \n')
cl = df.loc[df.polarity == 1, ['reviewText']].sample(5).values
for c in cl:
    print(c[0])
```

```
5 random reviews with the highest positive sentiment polarity:

Dress up a video with 27 misses and you have one hit, including the 28th dress sung to the tune of "Here Comes the Brid
e." Delightful entertainment!
This man should be postumously pardoned by the State of Montana. Excellent film. Forever one of the finest westerns mad
e.
Every time I think about trading this DVD in, I put it on an am reminded of why UNDERWORLD has been the best show I've e
ver seen in my life.
Excellent music.  Product was received in timely manner, and as advertised.
Lillies of the Field is one of the best movies by Sidney Poitier!! A thumbs up to ALL the nuns who played in it!!
```

In [ ]:

```
print('5 random reviews with the most neutral sentiment(zero) polarity: \n')
cl = df.loc[df.polarity == 0, ['reviewText']].sample(5).values
for c in cl:
    print(c[0])
```

```
5 random reviews with the most neutral sentiment(zero) polarity:

I had several of these horror flics marked to watch and deleted them all from my watch list. Not much of a story.
Bungled even by the likes of Gene Hackman,  Michael Rosenbaum at last defines Lex Luthor. Kristen becomes a star.  10-ye
ar hit status assured.
This movie reminds me of the movie &#34;Colors&#34;, only updated to present day. There is a bit of &#34;Adam 12&#34;, &
#34;Colors&#34;, &#34;Training Day&#34;, all mixed together and updated. Well done, and understated with the handheld ca
mera techniqued. Keeps you glued to the screen, all the while developing the relationship between the two LAPD cops on p
atrol. Definitely a must-see.
This was not a movie for me.  I actually could not finish watching it, just did not keep my interest.
This movie scared me to death 35 yrs ago and is still one of my favorites. Chilling is a word thrown around a lot in des
criptions of movies of this genre but in this case it fits. It doesnt get any better than this for sci-fi/horror of the
50's or any decade.
```

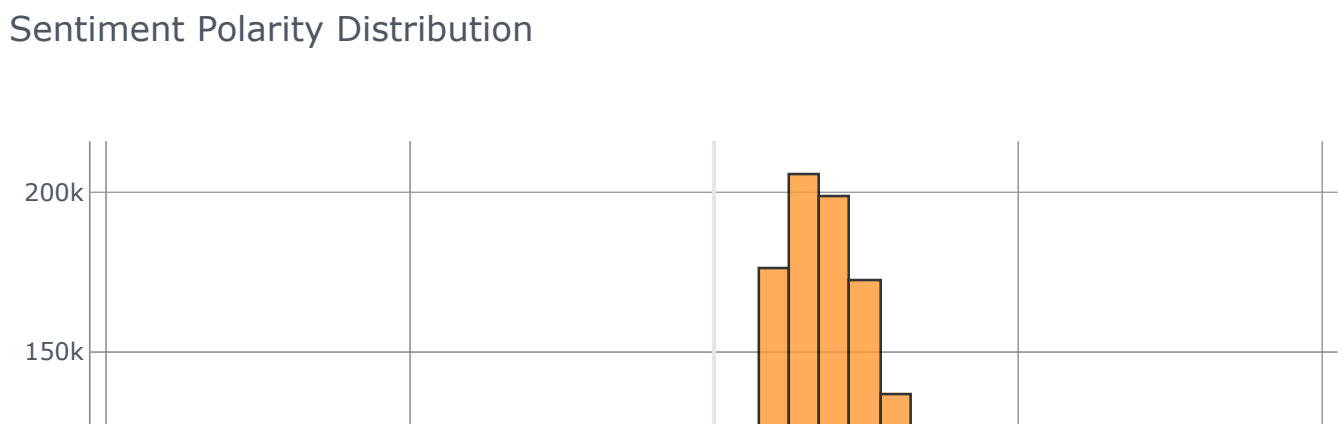
In [ ]:

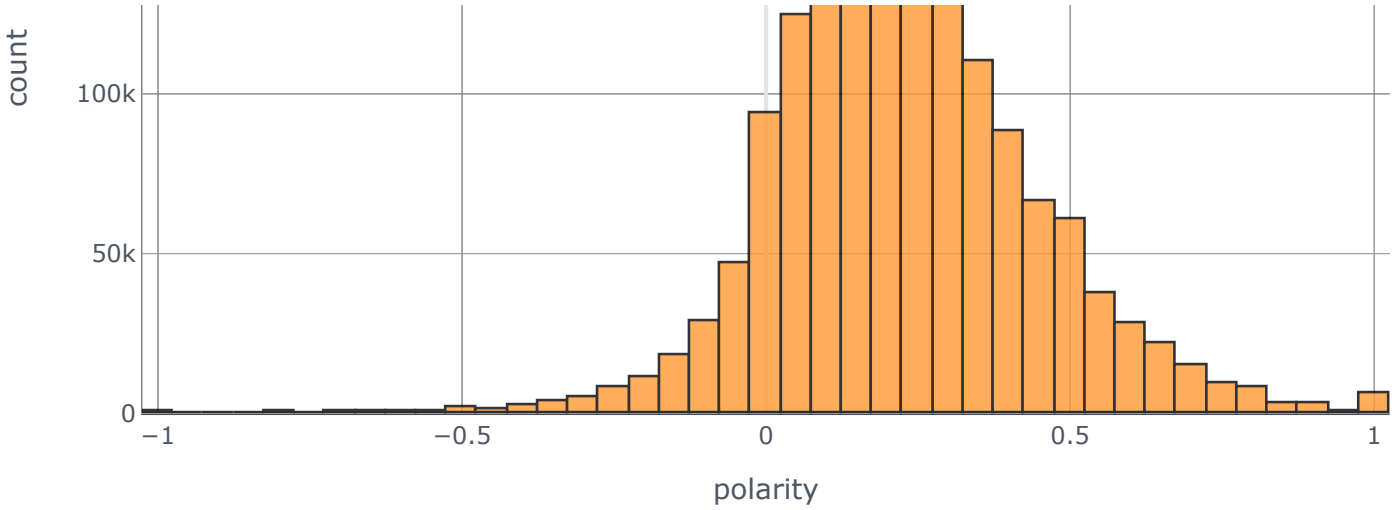
```
print('5 reviews with the most negative polarity: \n')
cl = df.loc[df.polarity == -0.5, ['reviewText']].sample(5).values
for c in cl:
    print(c[0])
```

```
5 reviews with the most negative polarity:

In this movie i through the friend such have realize sooner what he did to his friends it was awful a person had to lose
their life. But i would recommend this movie cause there is a lesson to be learned.
Not sure why people like this movie. This movie is not working. Just a sequence of scenes between dream /reality. Looks
like the director didn't know what to do. I got bored and disappointed.
once again Hollywood fails to deliver on the sequel&#8212;why do they bother.
I tried watching it but just was not into this dvd I was so bored with this movie I just did not like it.
it's as scary as it looks....Very intense.And the clown is just horrifying.I wouldn't let no kids watch this.it scared m
e.
```

In [ ]:

```
#Import Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

#Plotly Tools
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)
import plotly.graph_objs as go
import plotly.offline as offline
offline.init_notebook_mode()
from plotly import tools
import plotly.tools as tls
init_notebook_mode(connected=True)
import cufflinks as cf
cf.go_offline()
cf.set_config_file(offline=False, world_readable=True)

df['polarity'].iplot(
    kind='hist',
    bins=50,
    xTitle='polarity',
    linecolor='black',
    yTitle='count',
    title='Sentiment Polarity Distribution')
```

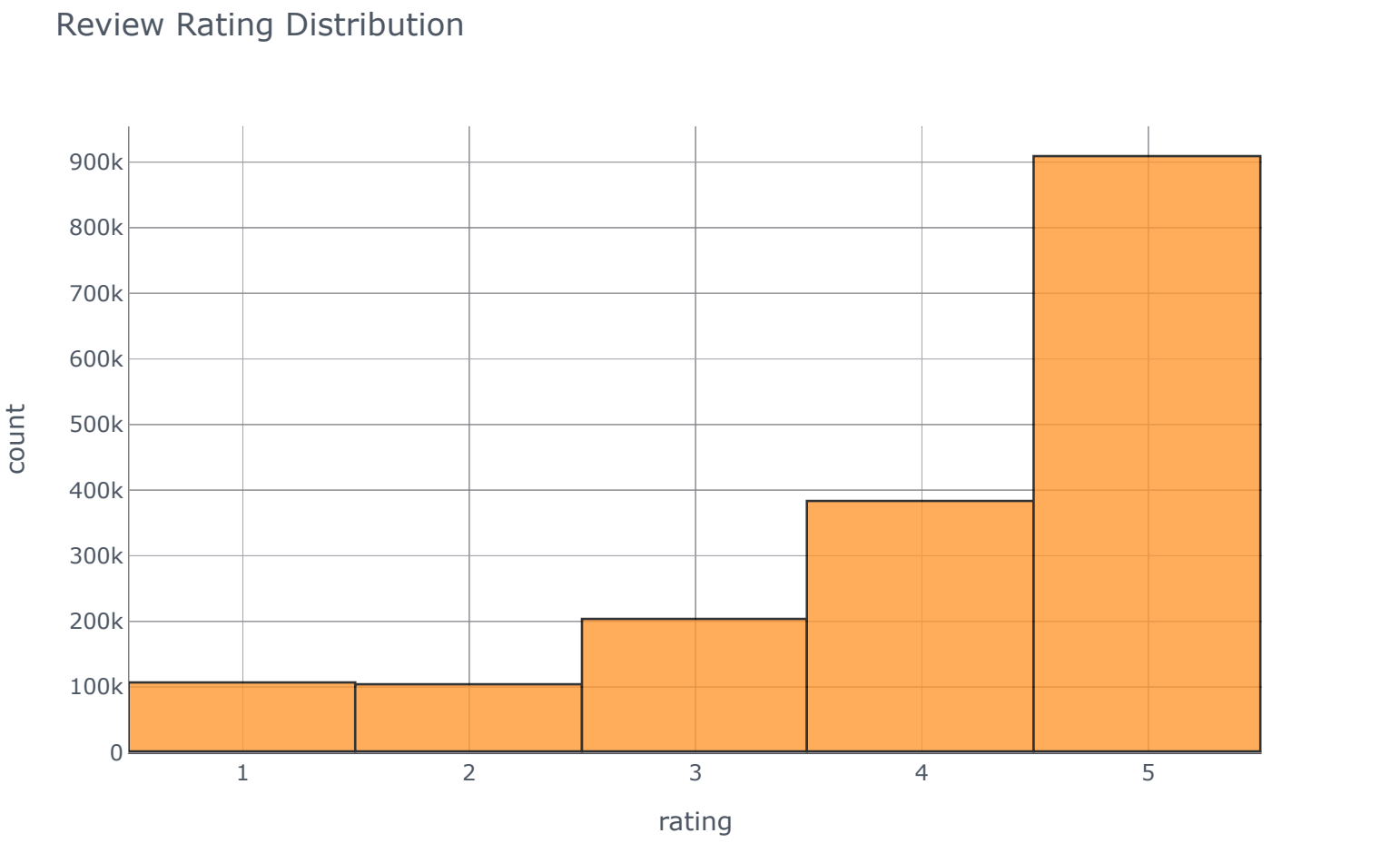




Export to plot.ly »

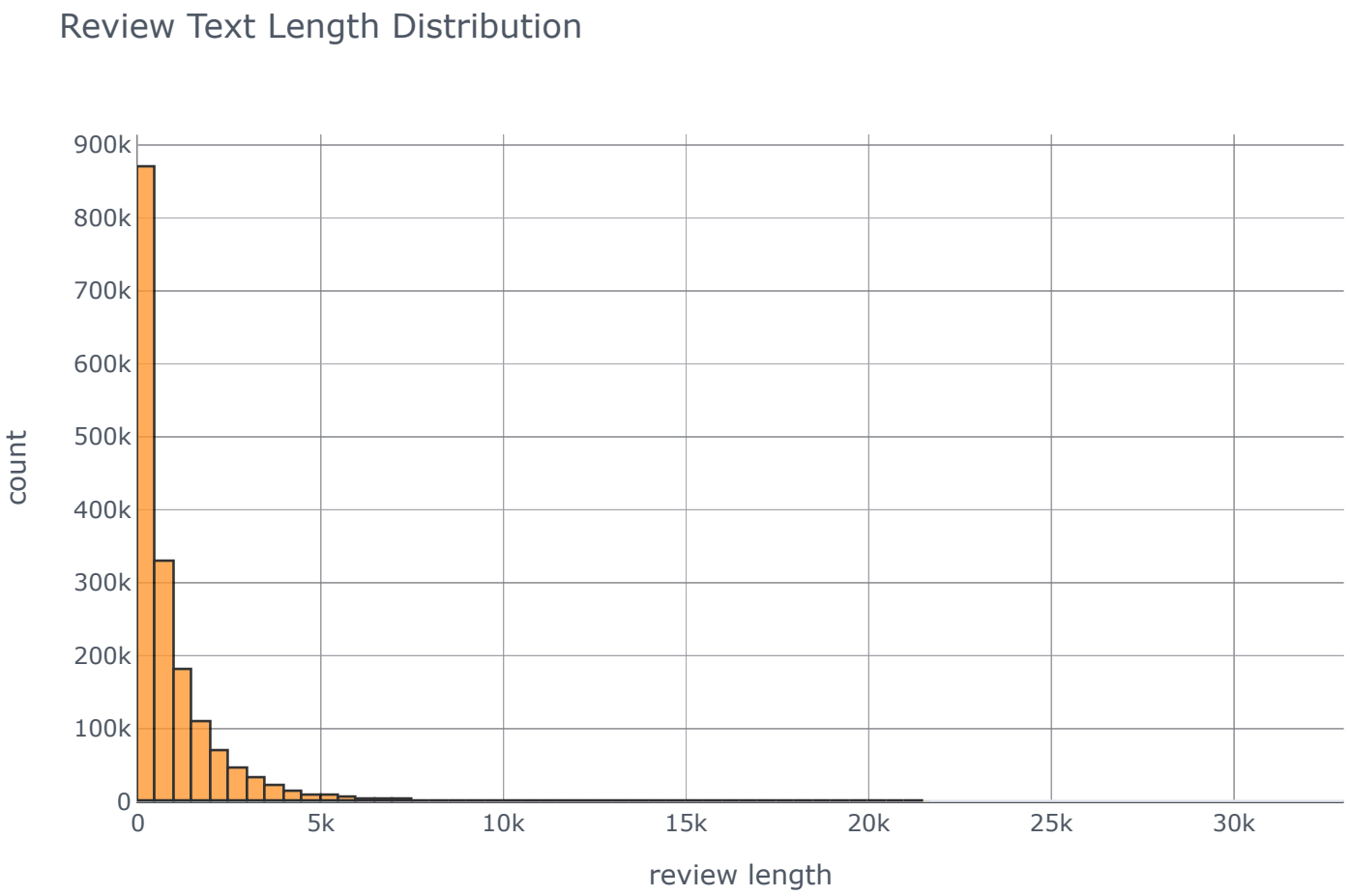
In [ ]:

```
df['overall'].iplot(
    kind='hist',
    xTitle='rating',
    linecolor='black',
    yTitle='count',
    title='Review Rating Distribution')
```



In [ ]:

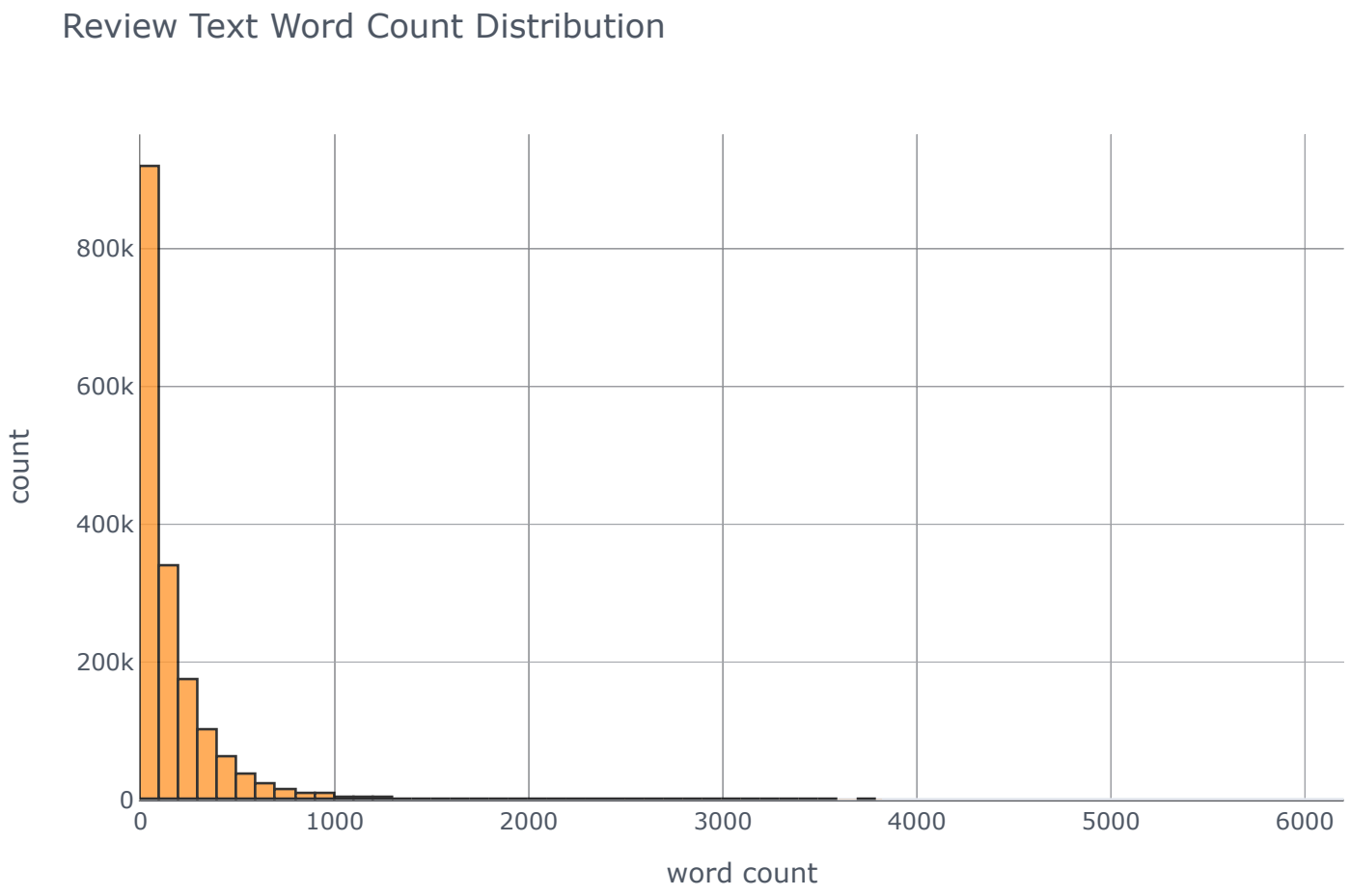
```
df['review_len'].iplot(
    kind='hist',
    bins=100,
    xTitle='review length',
    linecolor='black',
    yTitle='count',
    title='Review Text Length Distribution')
```



Export to plot.ly »

In [ ]:

```
df['word_count'].iplot(
    kind='hist',
    bins=100,
    xTitle='word count',
    linecolor='black',
    yTitle='count',
    title='Review Text Word Count Distribution')
```

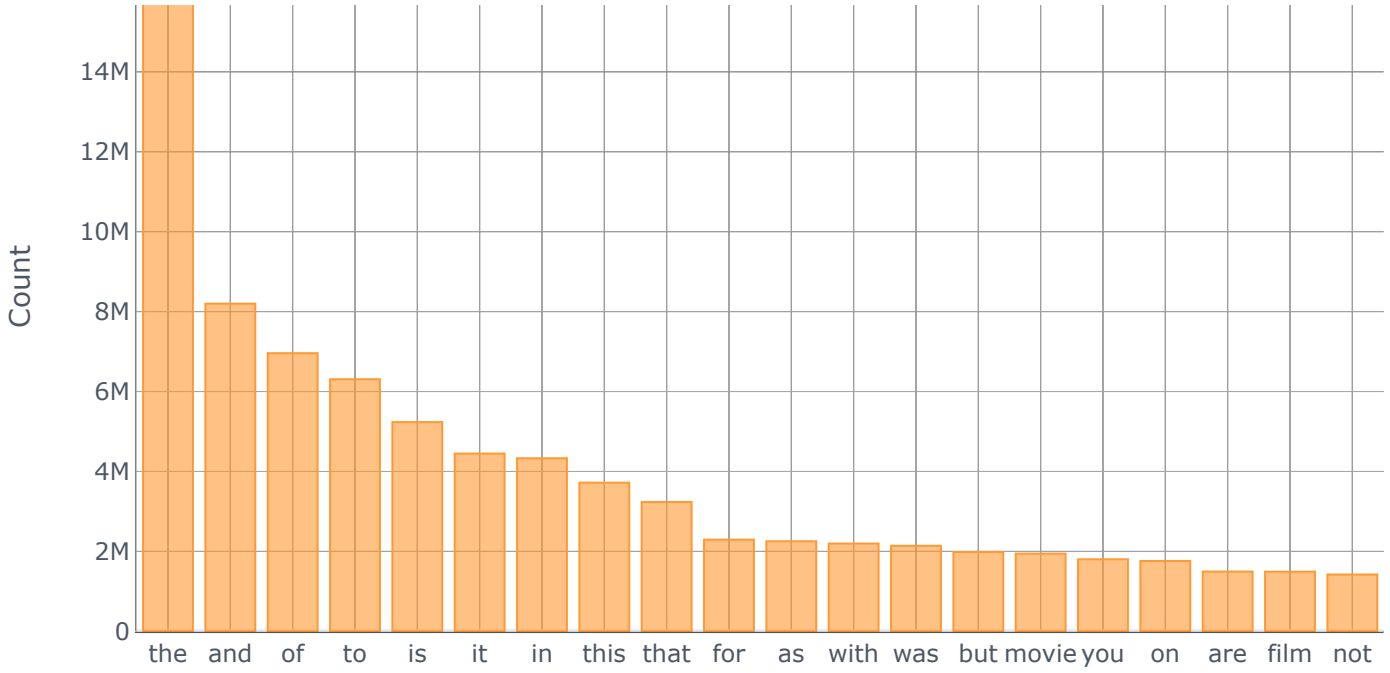


Export to plot.ly »

In [ ]:

```
def get_top_n_words(corpus, n=None):
    vec = CountVectorizer().fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
    return words_freq[:n]
common_words = get_top_n_words(df['reviewText'], 20)
for word, freq in common_words:
    print(word, freq)
df1 = pd.DataFrame(common_words, columns = ['reviewText' , 'count'])
df1.groupby('reviewText').sum()['count'].sort_values(ascending=False).iplot(
    kind='bar', yTitle='Count', linecolor='black', title='Top 20 words in review before removing stop words')
```

```
the 16399695
and 8198290
of 6959404
to 6307177
is 5235998
it 4447618
in 4330611
this 3720441
that 3235537
for 2295381
as 2254253
with 2196737
was 2140099
but 1984339
movie 1945059
you 1803016
on 1761735
are 1495803
film 1492831
not 1422246
```

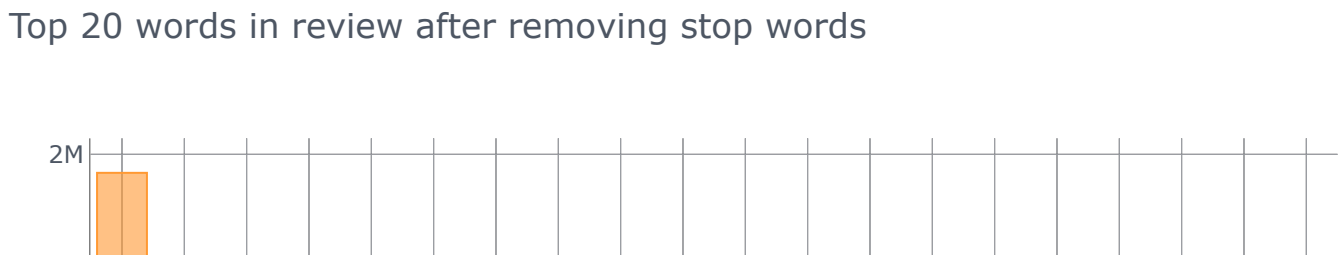


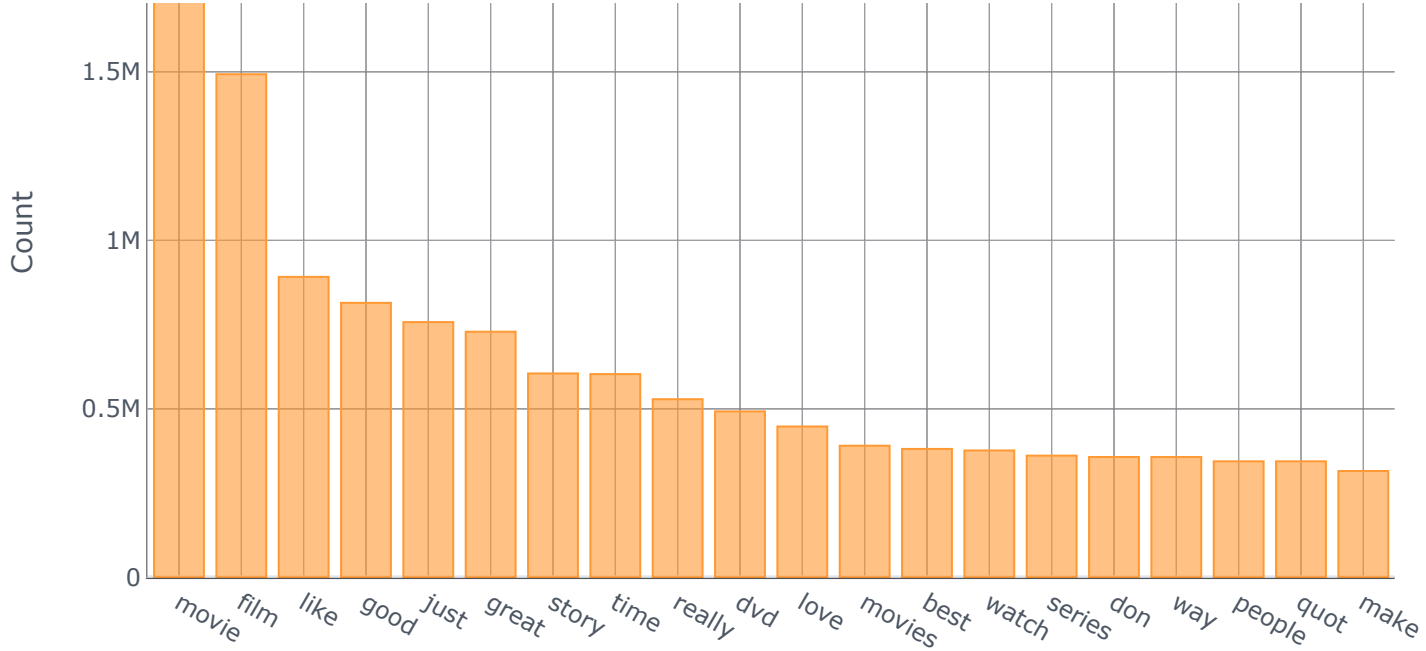
Export to plot.ly »

In [ ]:

```
def get_top_n_words(corpus, n=None):
    vec = CountVectorizer(stop_words = 'english').fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
    return words_freq[:n]
common_words = get_top_n_words(df['reviewText'], 20)
for word, freq in common_words:
    print(word, freq)
df2 = pd.DataFrame(common_words, columns = ['reviewText' , 'count'])
df2.groupby('reviewText').sum()['count'].sort_values(ascending=False).iplot(
    kind='bar', yTitle='Count', linecolor='black', title='Top 20 words in review after removing stop words')
```

```
movie 1945059
film 1492831
like 891611
good 814498
just 757456
great 728824
story 605037
time 603128
really 528601
dvd 492446
love 447648
movies 390588
best 381055
watch 376545
series 361202
don 357229
way 357008
people 344288
quot 344266
make 315549
```





Export to plot.ly »

In [ ]:

```
def get_top_n_trigram(corpus, n=None):
    vec = CountVectorizer(ngram_range=(3, 3), stop_words='english').fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
    return words_freq[:n]
common_words = get_top_n_trigram(df['reviewText'], 20)
for word, freq in common_words:
    print(word, freq)
df6 = pd.DataFrame(common_words, columns = ['reviewText' , 'count'])
df6.groupby('reviewText').sum()['count'].sort_values(ascending=False).iplot(
    kind='bar', yTitle='Count', linecolor='black', title='Top 20 trigrams in review after removing stop words')
```

In [ ]:

```
trace1 = go.Scatter(
    x=df['polarity'], y=df['overall'], mode='markers', name='points',
    marker=dict(color='rgb(102,0,0)', size=2, opacity=0.4)
)
trace2 = go.Histogram2dContour(
    x=df['polarity'], y=df['overall'], name='density', ncontours=20,
    colorscale='Hot', reversescale=True, showscale=False
)
trace3 = go.Histogram(
    x=df['polarity'], name='Sentiment polarity density',
    marker=dict(color='rgb(102,0,0)'),
    yaxis='y2'
)
trace4 = go.Histogram(
    y=df['overall'], name='Rating density', marker=dict(color='rgb(102,0,0)'),
    xaxis='x2'
)
data = [trace1, trace2, trace3, trace4]

layout = go.Layout(
    showlegend=False,
    autosize=False,
    width=600,
    height=550,
    xaxis=dict(
        domain=[0, 0.85],
        showgrid=False,
        zeroline=False
    ),
    yaxis=dict(
        domain=[0, 0.85],
        showgrid=False,
        zeroline=False
    ),
    margin=dict(
        t=50
    ),
    hovermode='closest',
    bargap=0,
    xaxis2=dict(
        domain=[0.85, 1],
        showgrid=False,
        zeroline=False
    ),
    yaxis2=dict(
```

```
        domain=[0.85, 1],
        showgrid=False,
        zeroline=False
    )
)

fig = go.Figure(data=data, layout=layout)
iplot(fig, filename='2dhistogram-2d-density-plot-subplots')
```

In [ ]:

```
1
```