# << 리뷰데이터를 긍정,부정을 나누어 워드클라우드 생성 >>

감성분석에서 만들었던 평점,리뷰 데이터로 긍정,부정을 나누어 워드 클라우드를 생성 긍정적인 리뷰에 많이 노출되는 단어를 추출하여 영화에서 어떤 부분이 좋았는지를 추측할 수 있다. 반대로 부정적인 댓글에서는 어느 점이 싫었는지를 알 수 있어, 개선에 도움을 받을 수 있다.

## 전체 과정

- step1. 데이터를 가져와서 스코어를 기준으로 긍정,부정 리뷰를 나눔
- step2. 나누어진 리뷰데이터를 형태서 분석하여 명사를 토큰화한다.
- step3. 토큰화 된 명사중에서 자주 나오는 50개의 단어를 확인
- step4. 해당 단어의 빈도수를 표현하는 차트 생성
- step5. 자주 나오는 500개의 단어로 워드 클라우드 생성

## step0. import

In [42]:
```
import nltk
import numpy as np
from konlpy.tag import Okt; t = Okt()
import pandas as pd
import platform
import matplotlib.pyplot as plt

%matplotlib inline

path = "c:/Windows/Fonts/malgun.ttf"
from matplotlib import font_manager, rc
if platform.system() == 'Darwin':
    rc('font', family='AppleGothic')
elif platform.system() == 'Windows':
    font_name = font_manager.FontProperties(fname=path).get_name()
    rc('font', family=font_name)
else:
    print('Unknown system... sorry~~~~')

plt.rcParams['axes.unicode_minus'] = False
```

## step1. 데이터를 가져와서 스코어를 기준으로 긍정,부정 리뷰를 나눔

In [2]:
```
df=pd.read_csv('star_score.csv') # 앞서 만들었던 평점과 리뷰데이터를 이용
```

In [3]:
```
df1=df[df.score<5] #0~4점은 부정
df2=df[df.score>7] #8~10점은 긍정
```

In [4]:
```
df1=df1['text'] # 그 중에서 긍정 리뷰만 추출
df2=df2['text'] # 부정 리뷰 추출
```

In [5]:
```
df1
```

Out[5]:
```
43                                          극장에서 보면 돈 아까운영화
45                                           재미도 없고 감동도 없음
49       정말 진심으로 왜 재밌다하는지 모르겠는 영화..한국식으로 감동포인트 억지로 만든 것...
51                                          하 ㅠㅠ 한국영화수준 참
56       얼마나 국내 영화 알바들이 많은지 알겠다.. 어거지 개연성 떨어지고.. 졸라 재미없...
                                ...
9748                                여니여니귀여니한테 배워야 할 듯...
9749                                      rkskekfkakqktk
10074                    와 더럽게 재미없네.. 도대체 저 평점이 왜 나오는거지;
10105          개연성이 이렇게 떨어지는 이야기가 별점9 점이상이라니  놀랍고실망스럽다
10230                              노래가 좋고.. 쟈스민이 예쁘다 정도..
Name: text, Length: 1688, dtype: object
```

In [6]:
```
df2
```

Out[6]:
```
0                              엑시트가 재밌으면 추천 사자가 재밌으면 비추천
1                                    교훈 : 옥상문을 열고 다니자
2       솔직히 정말 재밌게 봤습니다  알바아니에요 저 닉네임 클릭하면 평점 매긴 영화들 나...
3                                  여태 이런 느낌의 재난영화는 없었다
4       윤아우는거 졸귀졸웃 ㅋㅋㅋㅋ 조정석은 뭐 완전 찰떡 조정석아니었음 못살렸을 것같은 느낌
                                ...
10245   여주 진심 연기잘함ㅠㅠ 이영화는 꼭 극장에서 보셔야합니다!!! 실사인데도 진짜 재밌...
10246   디즈니는 사람들을 동화 속으로 초대하는 마법사다. 정말 눈물 나게 아름다운 영화! ...
10247                                    안보면 절대 후회합니다
10248                                고작 10점? 10억점으로도 부족해
10249   태어나서 생전 처음으로 재관람 하고싶어진 영화. 진짜 모든 순간들을 너무나도 아름다...
Name: text, Length: 8170, dtype: object
```

## step2. 나누어진 리뷰데이터를 형태소 분석하여 명사를 토큰화한다.

In [26]:
```
pos= ''

for each_line in df2[:4000]:
    pos = pos + each_line + '₩n'
```

In [47]:
```
tokens_pos = t.nouns(pos) #형태소 분석 Okt
tokens_pos[0:10]
```

Out[47]: ['시트', '추천', '사자', '비', '추천', '교훈', '옥상', '문', '정말', '알바']

In [28]:
```
neg = ''

for each_line in df1[:4000]:
    neg = neg + each_line + '₩n'
```

In [29]:
```
tokens_neg = t.nouns(neg)
tokens_neg[0:10]
```

Out[29]: ['극장', '돈', '영화', '재미', '감동', '정말', '진심', '왜', '영화', '한국']

In [30]:
```
po = nltk.Text(tokens_pos, name='영화')
print(len(po.tokens))
print(len(set(po.tokens)))
```

```
29532
3050
```

In [31]:
```
ne = nltk.Text(tokens_neg, name='영화')
print(len(ne.tokens))
print(len(set(ne.tokens)))
```

```
14533
1883
```

## step3. 토큰화 된 명사중에서 자주 나오는 50개의 단어를 확인

In [32]:
```
pos_data=po.vocab().most_common(50)
pos_data
```

Out[32]:
```
[('영화', 1369),
 ('관람객', 703),
 ('진짜', 530),
 ('윤아', 461),
 ('조정석', 451),
 ('연기', 443),
 ('따따', 257),
 ('보고', 249),
 ('배우', 238),
 ('추천', 219),
 ('액션', 214),
 ('최고', 192),
 ('가족', 190),
 ('정말', 185),
 ('꼭', 183),
 ('더', 182),
 ('생각', 173),
 ('시트', 162),
 ('볼', 158),
 ('감동', 156),
 ('시간', 152),
 ('재난영화', 144),
 ('시사회', 139),
 ('역시', 137),
 ('이', 137),
 ('보기', 133),
 ('재난', 127),
 ('것', 124),
 ('거', 123),
 ('완전', 120),
 ('중간', 118),
 ('수', 117),
 ('임윤아', 116),
 ('한국', 113),
 ('웃음', 112),
 ('시리즈', 112),
 ('여름', 111),
 ('때', 111),
 ('분노', 111),
 ('웃기', 108),
 ('기대', 108),
 ('손', 108),
 ('그냥', 108),
 ('긴장감', 107),
 ('사람', 105),
 ('스릴', 105),
 ('안', 102),
 ('함', 102),
 ('여주', 102),
 ('빡빡이', 102)]
```

In [33]:
```
neg_data=ne.vocab().most_common(50)
neg_data
```

Out[33]:
```
[('영화', 736),
 ('평점', 317),
 ('용남', 315),
 ('진짜', 277),
 ('점', 205),
 ('알바', 182),
 ('이', 164),
 ('돈', 154),
 ('왜', 152),
 ('질주', 146),
 ('분노', 131),
 ('감동', 123),
 ('보고', 123),
 ('분노의질주', 121),
 ('그냥', 118),
 ('최악', 113),
 ('말', 108),
 ('시리즈', 106),
 ('정말', 105),
 ('스토리', 97),
 ('재미', 90),
 ('개연', 88),
 ('거', 86),
 ('노잼', 83),
 ('웃음', 81),
 ('댓글', 80),
 ('한국영', 79),
 ('수준', 79),
 ('중간', 77),
 ('임', 71),
 ('억지', 71),
 ('액션', 68),
 ('배우', 68),
 ('이건', 66),
 ('진심', 62),
 ('사람', 62),
 ('보지', 61),
 ('것', 60),
 ('시간', 60),
 ('내', 59),
 ('별로', 58),
 ('생각', 57),
 ('일본', 56),
 ('듯', 55),
 ('정도', 53),
 ('좀', 53),
 ('느낌', 51),
 ('무슨', 51),
 ('절대', 50),
 ('지금', 50)]
```

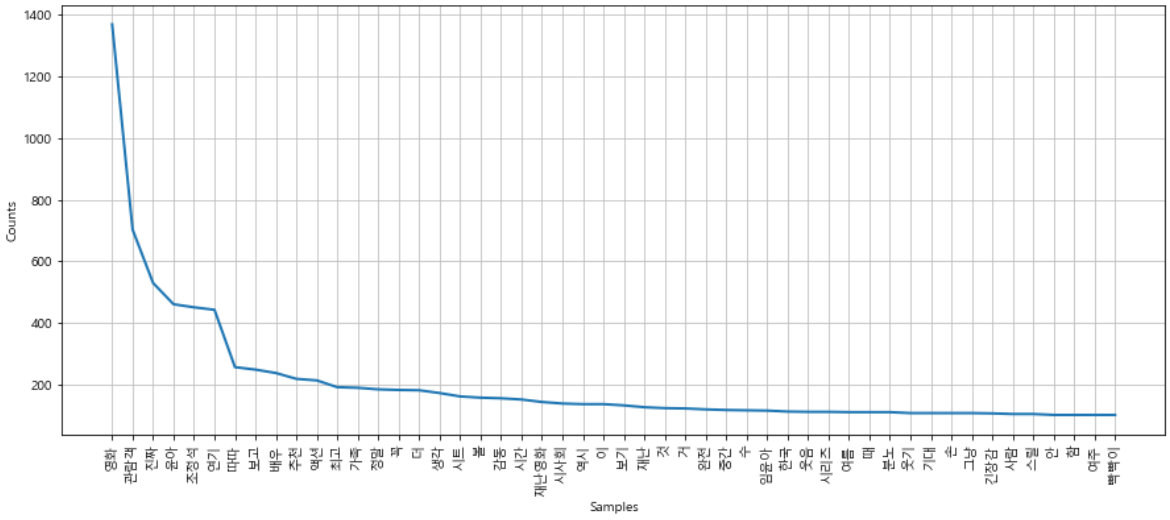
## step4. 해당 단어의 빈도수를 표현하는 차트 생성

In [37]:
```
from wordcloud import WordCloud, STOPWORDS
from PIL import Image
import platform
import matplotlib.pyplot as plt
```

In [38]:
```
po.similar('영화') #nltk 안에 있는 주어진 단어와 비슷한 환경에서 쓰인 단어를 추출
```

조정석 관람객 윤아 최고 진짜 연기 보고 추천 정말 따따 생각 더 시간 배우 액션 장난 재난영화 완전 웃기 가족

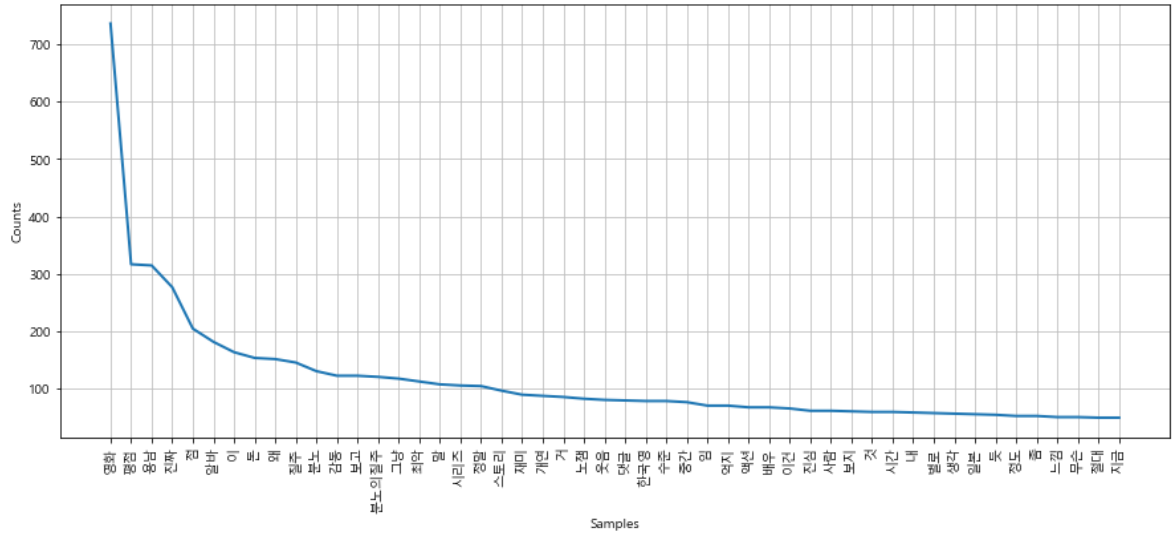
In [43]:
```
plt.figure(figsize=(15,6))
po.plot(50)
plt.show() #긍정 리뷰에서 많이 나오는 단어
```




In [16]:
```
ne.similar('영화')
```

왜 진짜 재난영화 내용 감동 최악 알바 평점 용남 댓글알바 임 추천 이해 절대 중간 관객 관람객 산업 무슨 좀

In [45]:
```
plt.figure(figsize=(15,6))
ne.plot(50)
plt.show() #부정 리뷰에서 많이 나오는 단어
```




## step5. 자주 나오는 500개의 단어로 워드 클라우드 생성

In [20]:
```
mask = np.array(Image.open('popcorn.png'))
from wordcloud import ImageColorGenerator
image_colors = ImageColorGenerator(mask)

image_colors
```

Out[20]: <wordcloud.color_from_image.ImageColorGenerator at 0x1e3ade3c048>

In [21]:
```
pos_data = pos.vocab().most_common(500)
# for win : font_path='c:/Windows/Fonts/malgun.ttf'
wordcloud = WordCloud(font_path='c:/Windows/Fonts/jalnan.ttf',
                relative_scaling = 0.1, mask=mask,
                background_color = 'white',
                min_font_size=1,
                max_font_size=100).generate_from_frequencies(dict(pos_data))

default_colors = wordcloud.to_array()
```

In [22]:
```
plt.figure(figsize=(12,12))
plt.imshow(wordcloud.recolor(color_func=image_colors), interpolation='bilinear')
plt.axis('off')
plt.show() #긍정 워드클라우드
```


In [23]:
```
neg_data = neg.vocab().most_common(500)
# for win : font_path='c:/Windows/Fonts/malgun.ttf'
wordcloud = WordCloud(font_path='c:/Windows/Fonts/jalnan.ttf',
                relative_scaling = 0.1, mask=mask,
                background_color = 'black',
                min_font_size=1,
                max_font_size=100).generate_from_frequencies(dict(neg_data))

default_colors = wordcloud.to_array()
```

In [24]:
```
plt.figure(figsize=(12,12))
plt.imshow(wordcloud.recolor(color_func=image_colors), interpolation='bilinear')
plt.axis('off')
plt.show() #부정 워드클라우드
```