Sentiment Analysis

In [5]:

```
import pandas as pd

data = pd.read_csv('0630.txt', header=None, names=["no", "평점", "제목", "리뷰", "작성일"])
reviews = data.drop_duplicates(['리뷰'])
data.head()
```

Out[5]:

	no	평점	제목	리뷰	작성일
0	15862028	10	존 윅 3: 파라벨룸	스토리 운운하지 마라. 알고 보는거잖아? 말이 필요없다. 액션 진짜 최고다.	19.06.30
1	15862027	10	존 윅 3: 파라벨룸	난 되려 뭔가 느린 액션이 더 현실감있게 다가오던데ㅋ 2시간이 후딱지나간게 아쉬울정...	19.06.30
2	15862026	7	돈	ㅎㅎ 알바 마니 있네.저는 7점요.증권용어 좀 이해안되고 VOD로는 볼만할듯.긴장감...	19.06.30
3	15862025	10	내 이야기!!	허그라도 한번하지 ㅠ 여주만 봐도 기분 좋은 영화 ㅋ	19.06.30
4	15862024	10	롱 리브 더 킹: 목포 영웅	그래 본 한국영화 중 제일 재밌게 본 것 같아요. 배우들 연기 최고입니다.	19.06.30

In [6]:

```
reviews.count()
```

Out[6]:

```
no      9915
평점      9915
제목      9915
리뷰      9915
작성일     9915
dtype: int64
```

In [7]:

```
reviews.groupby(['평점']).count()
```

Out[7]:

평점	no	제목	리뷰	작성일
1	891	891	891	891
2	427	427	427	427
3	84	84	84	84
4	259	259	259	259
5	201	201	201	201
6	583	583	583	583
7	398	398	398	398
8	1100	1100	1100	1100
9	699	699	699	699
10	5273	5273	5273	5273

In [8]:

```
positive = reviews[(reviews['평점'] > 7) & (reviews['평점'] < 10)]
negative = reviews[(reviews['평점'] < 5)]
posN = len(positive)
negN = len(negative)
N = len(reviews)
```

In [9]:

```
from collections import Counter, defaultdict
from konlpy.tag import Komoran

ma = Komoran()
```

In [10]:

```
posDTM = defaultdict(Counter)
for i,review in enumerate(positive['리뷰']):
    for pos in ma.pos(review):
        if len(pos[0]) > 1:
            posDTM[i][pos] += 1

negDTM = defaultdict(Counter)
for i,review in enumerate(negative['리뷰']):
    for pos in ma.pos(review):
        if len(pos[0]) > 1:
            negDTM[i][pos] += 1
```

In [11]:

```
posTDM = defaultdict(Counter)
for i,posList in posDTM.items():
    for t,freq in posList.items():
        posTDM[t][i] = freq

negTDM = defaultdict(Counter)
for i,posList in negDTM.items():
    for t,freq in posList.items():
        negTDM[t][i] = freq
```

In [12]:

```
posVoca = list(set(posTDM.keys()))
negVoca = list(set(negTDM.keys()))
```

In [13]:

```
posPOS = defaultdict(Counter)
for t in posVoca:
    posPOS[t[1]][t[0]] += sum(posTDM[t].values())

negPOS = defaultdict(Counter)
for t in negVoca:
    negPOS[t[1]][t[0]] += sum(negTDM[t].values())
```

In [15]:

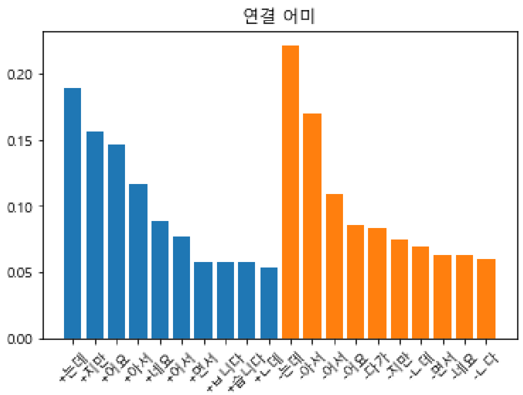
```
list(zip(posPOS['NNG'].most_common()[:20], negPOS['NNG'].most_common()[:20]))
```

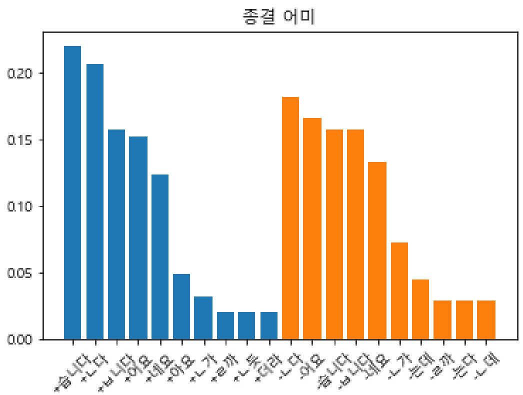
Out[15]:

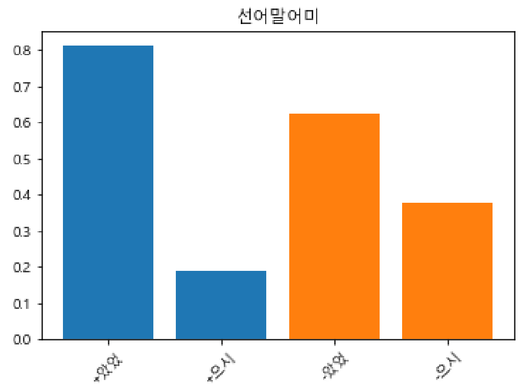
```
[(('영화', 417), ('영화', 529)),
 (('생각', 119), ('평점', 108)),
 (('연기', 94), ('내용', 92)),
 (('기대', 75), ('사람', 75)),
 (('사람', 72), ('개연', 73)),
 (('느낌', 60), ('감독', 72)),
 (('마지막', 57), ('연기', 70)),
 (('내용', 53), ('최악', 66)),
 (('최고', 51), ('생각', 54)),
 (('감동', 46), ('기대', 54)),
 (('배우', 42), ('느낌', 49)),
 (('시간', 36), ('시간', 48)),
 (('장면', 33), ('배우', 47)),
 (('결말', 32), ('처음', 44)),
 (('처음', 31), ('중간', 44)),
 (('평점', 30), ('전개', 37)),
 (('작품', 29), ('수준', 33)),
 (('전개', 29), ('재미', 32)),
 (('재미', 27), ('결말', 25)),
 (('이해', 27), ('기분', 25))]
```

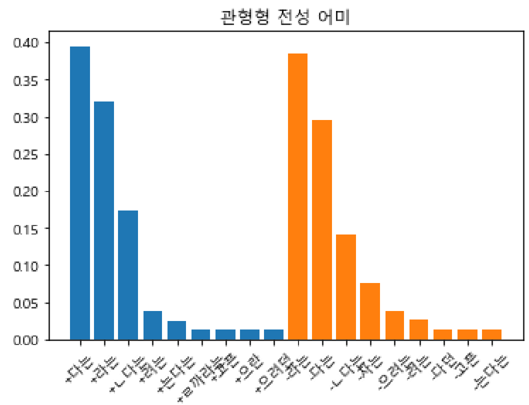
In [18]:

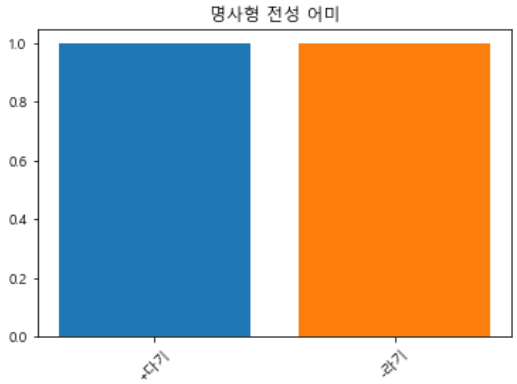
```
import matplotlib.pyplot as plt
from matplotlib import rc,font_manager

font = font_manager.FontProperties(fname='C:/Windows/Fonts/malgun.ttf').get_name()

rc('font', family=font)

for pos, kor in ma.tagset.items():
    _pos = posPOS[pos].most_common()[:10]
    _neg = negPOS[pos].most_common()[:10]
    _posSum = sum([_[1] for _ in _pos])
    _negSum = sum([_[1] for _ in _neg])
    plt.bar(['+'+_[0] for _ in _pos], [_[1]/_posSum for _ in _pos])
    plt.bar(['-'+_[0] for _ in _neg], [_[1]/_negSum for _ in _neg])
    plt.title(kor)
    plt.xticks(rotation=45)
    plt.show()
```

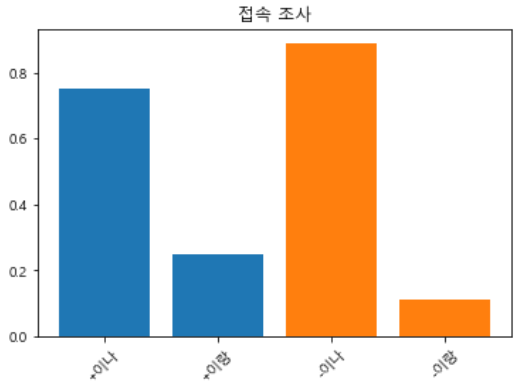

연결 어미
+는데
+지만
+어요
+아서
+네요
+어서
+면서
+ㅂ니다
+습니다
+ㄴ데
-는데
-아서
-어서
-어요
-다가
-지만
-ㄴ데
-면서
-네요
-ㄴ다

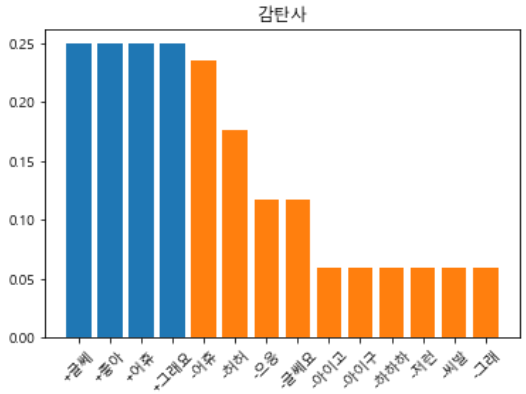

종결 어미

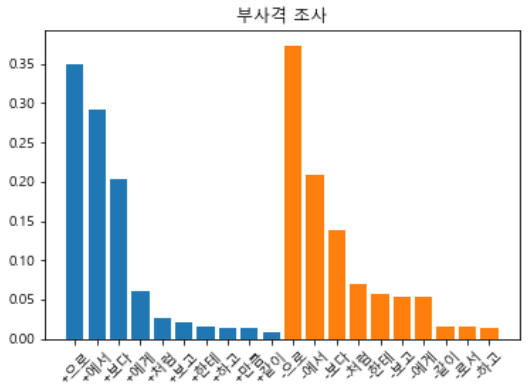

선어말어미
+았었
+으시
-았었
-으시

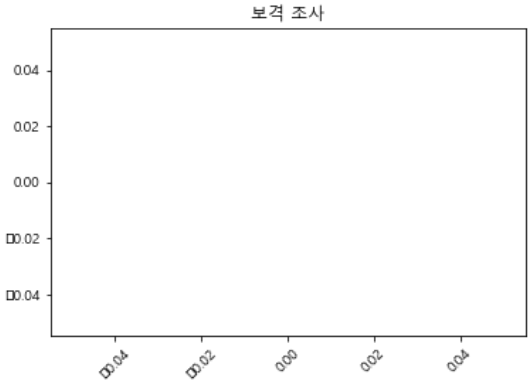

관형형 전성 어미

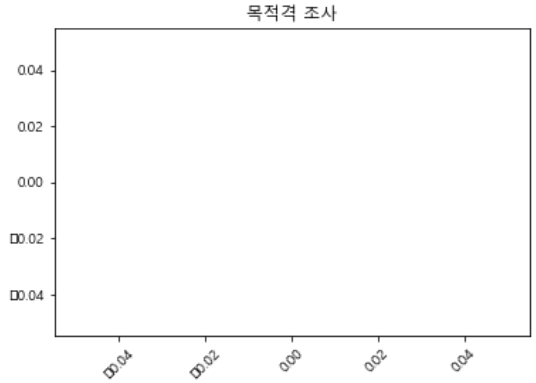

명사형 전성 어미
1.0
0.8
0.6
0.4
0.2
0.0
+다기
_다기

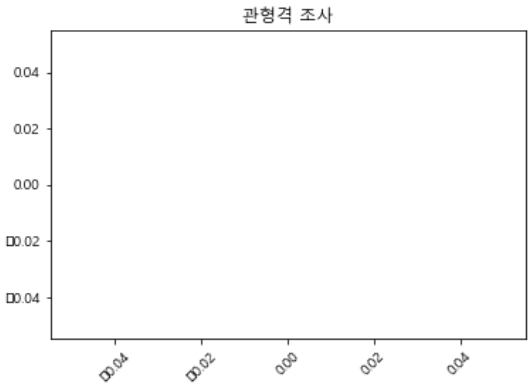

접속 조사
0.8
0.6
0.4
0.2
0.0
+이나
+이랑
_이나
_이랑

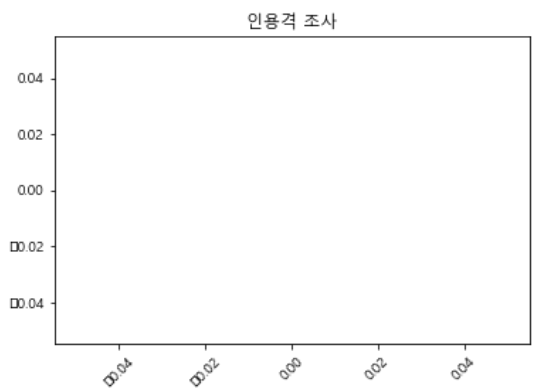

감탄사
0.25
0.20
0.15
0.10
0.05
0.00
+글쎄
+좋아
+어휴
+그래요
_어휴
_허허
_으응
_글쎄요
_아이고
_아이구
_하하하
_저런
_씨발
_그래

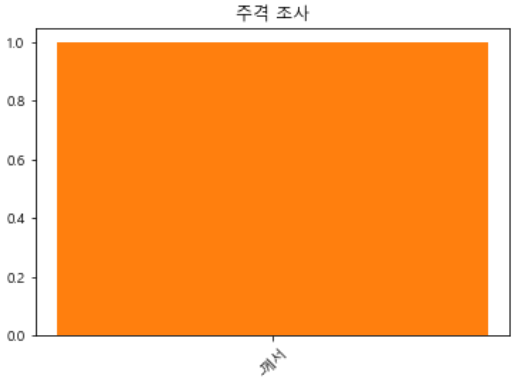

부사격 조사
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
+으로
+에서
+보다
+에게
+처럼
+보고
+한테
+하고
+만큼
+같이
_으로
_에서
_보다
_처럼
_한테
_보고
_에게
_같이
_로서
_하고

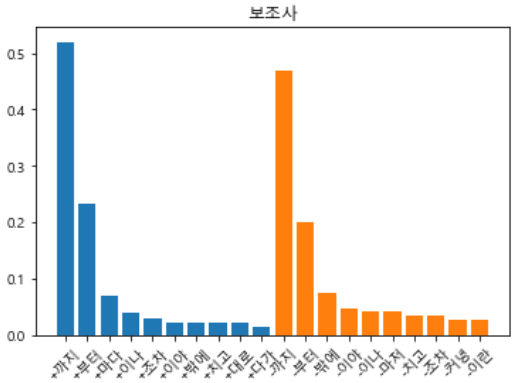

보격 조사
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

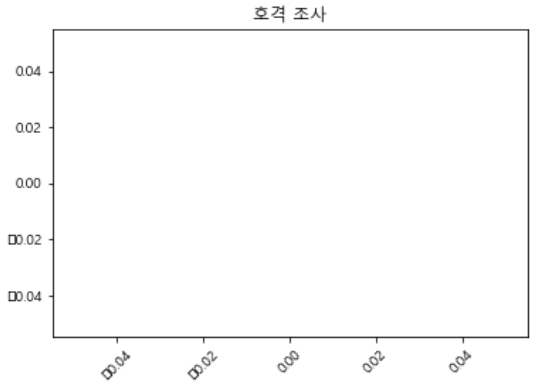

목적격 조사
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

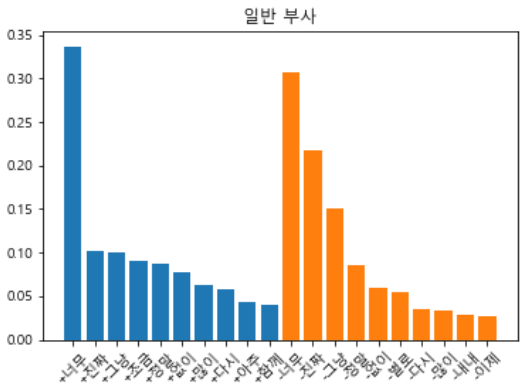

관형격 조사
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

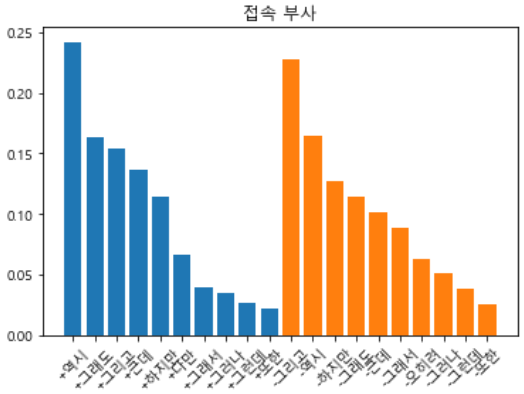

인용격 조사
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

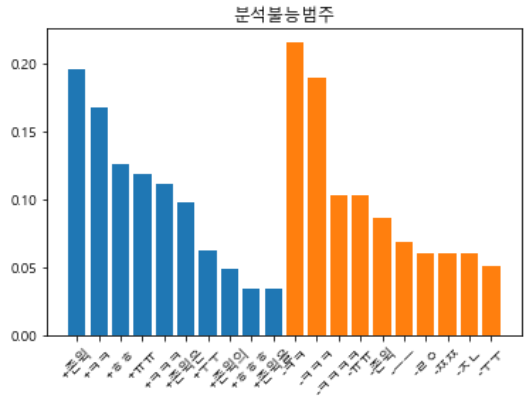

주격 조사
1.0
0.8
0.6
0.4
0.2
0.0
께서

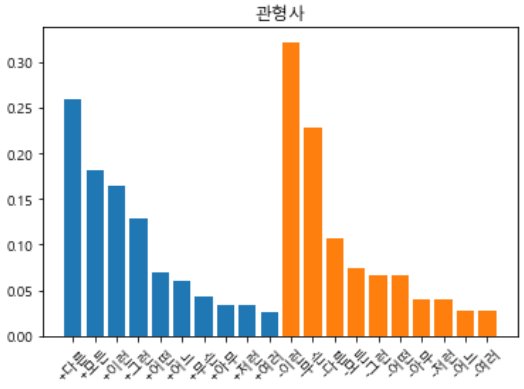

보조사
0.5
0.4
0.3
0.2
0.1
0.0
+까지
+부터
+마다
+이나
+조차
+이야
+밖에
+치고
+대로
+다가
-까지
-부터
-밖에
-이야
-이나
-마저
-치고
-조차
-커녕
-이란

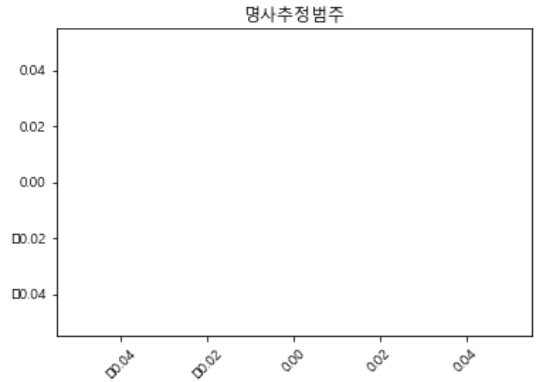

호격 조사
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

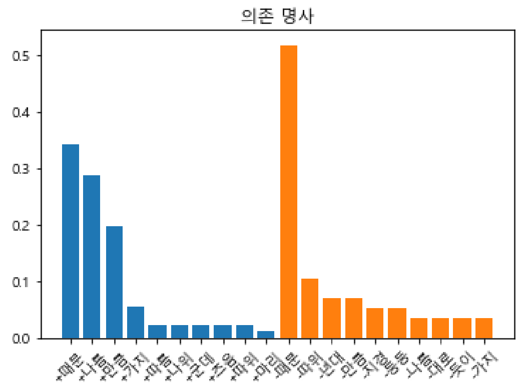

일반 부사
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
+너무
+진짜
+그냥
+정말
+참
+많이
+다시
+아주
+함께
-너무
-진짜
-그냥
-정말
-참
-많이
-다시
-아주
-함께
-이제

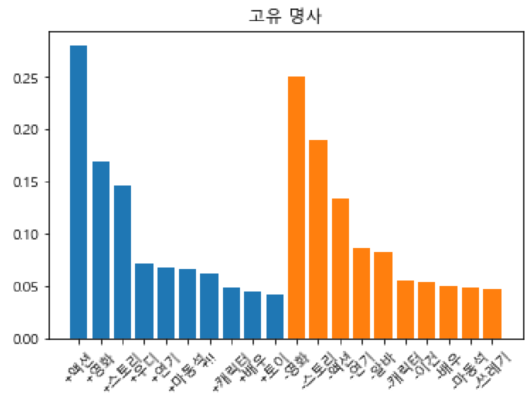

접속 부사
0.25
0.20
0.15
0.10
0.05
0.00

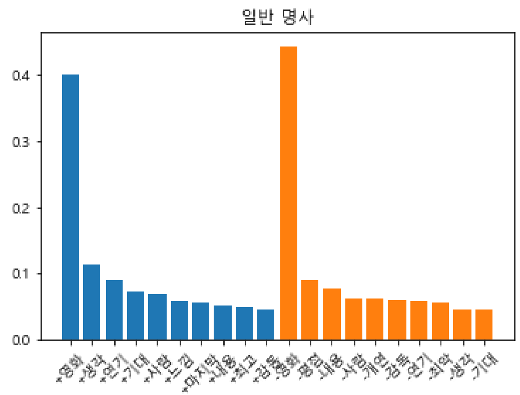

분석불능범주
0.20
0.15
0.10
0.05
0.00

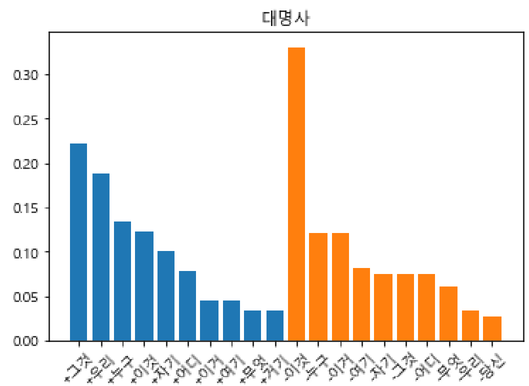

관형사
0.30
0.25
0.20
0.15
0.10
0.05
0.00

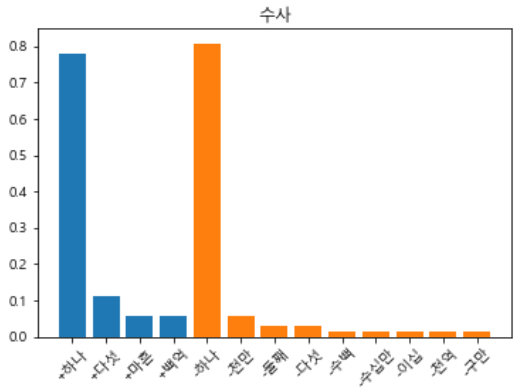

명사추정범주
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

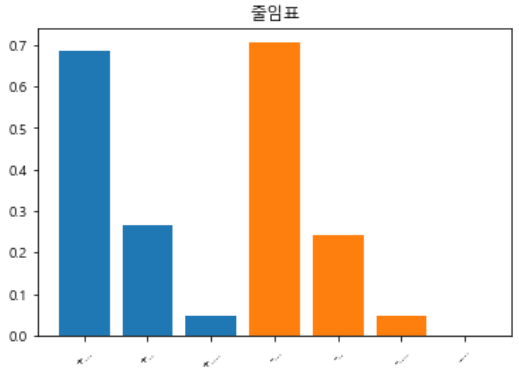

의존 명사
0.0
0.1
0.2
0.3
0.4
0.5

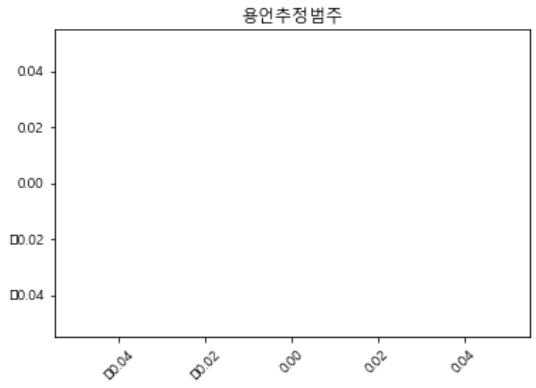

고유 명사
0.00
0.05
0.10
0.15
0.20
0.25

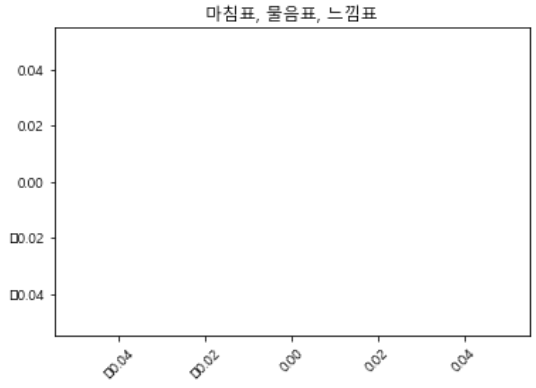

일반 명사
0.0
0.1
0.2
0.3
0.4

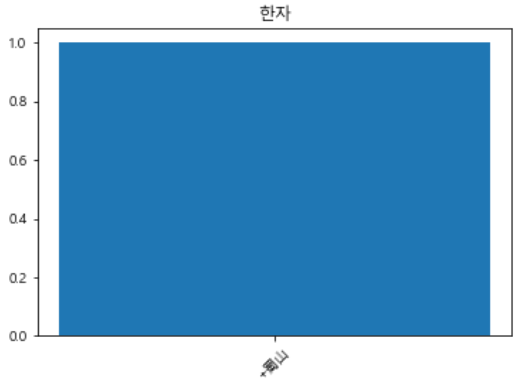

대명사
0.00
0.05
0.10
0.15
0.20
0.25
0.30

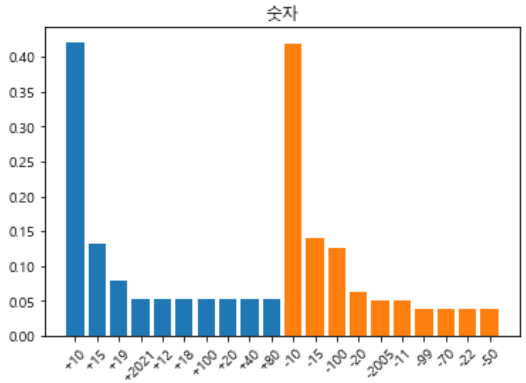

수사
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
+하나
+다섯
+마흔
+백억
하나
천만
둘째
다섯
수백
육십만
이십
천억
구만

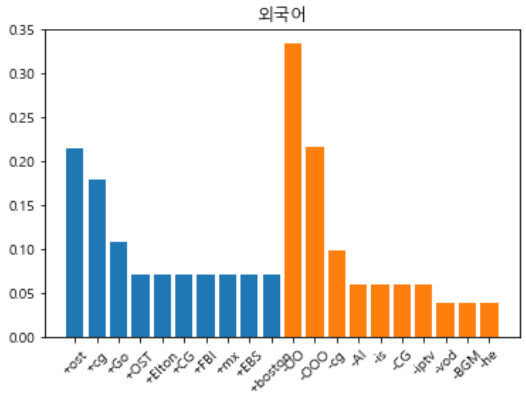

줄임표
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0

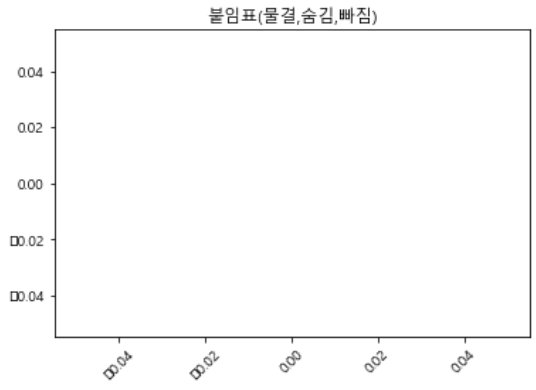

용언추정범주
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

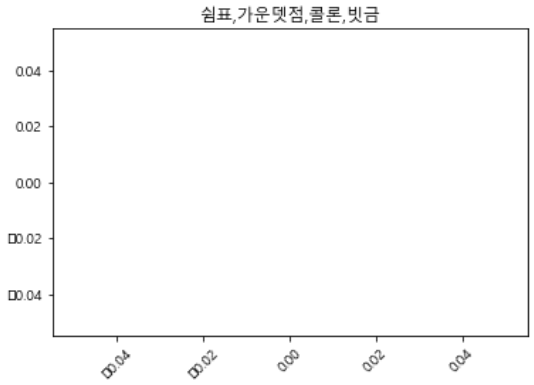

마침표, 물음표, 느낌표
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

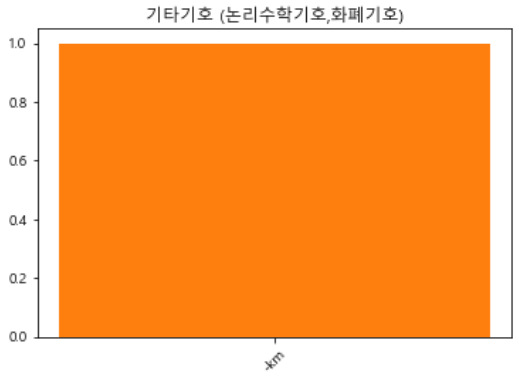

한자
1.0
0.8
0.6
0.4
0.2
0.0
+출시

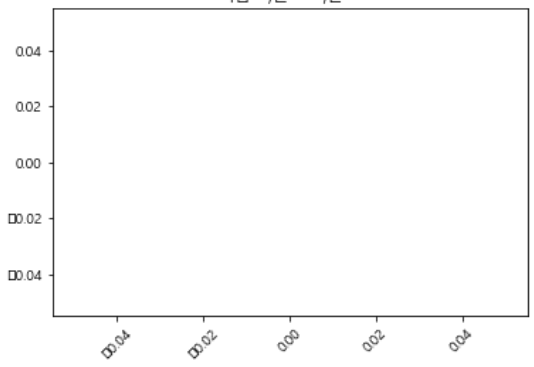

숫자
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
+10
+15
+19
+2021
+12
+18
+100
+20
+40
+80
-10
-15
-100
-20
-2005
-11
-99
-70
-22
-50

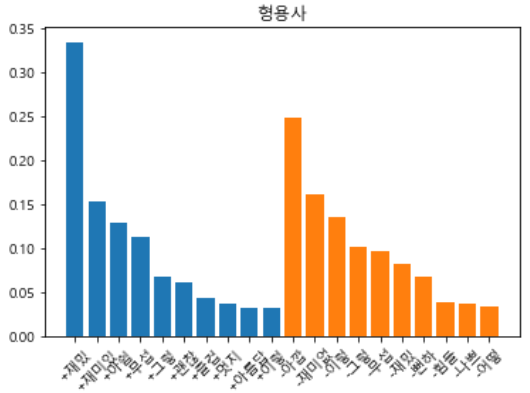

외국어
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
+ost
+cg
+Go
+OST
+Elton
+CG
+FBI
+mx
+EBS
+boston
-OO
-OOO
-cg
-AI
-s
-CG
-iptv
-vod
-BGM
-he

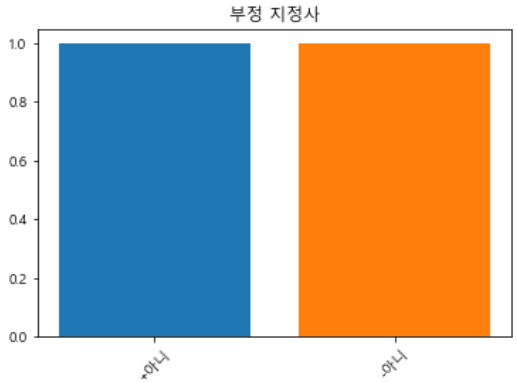

붙임표(물결,숨김,빠짐)
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

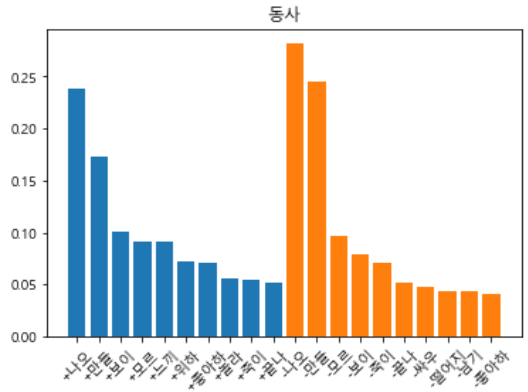

쉼표,가운뎃점,콤론,빗금
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

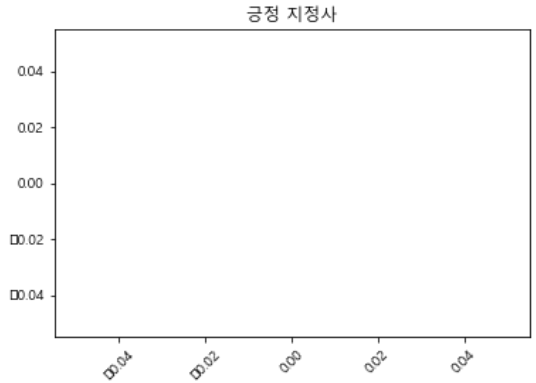

기타기호 (논리수학기호,화폐기호)
1.0
0.8
0.6
0.4
0.2
0.0
-km

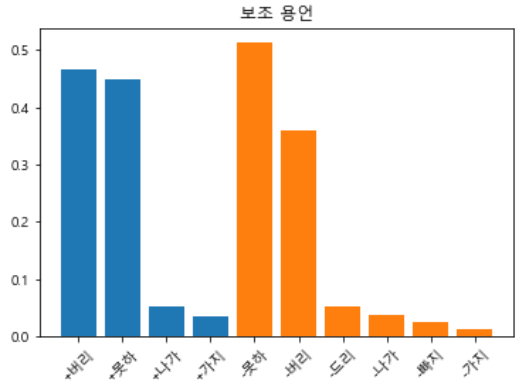

따옴표,괄호표,줄표
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

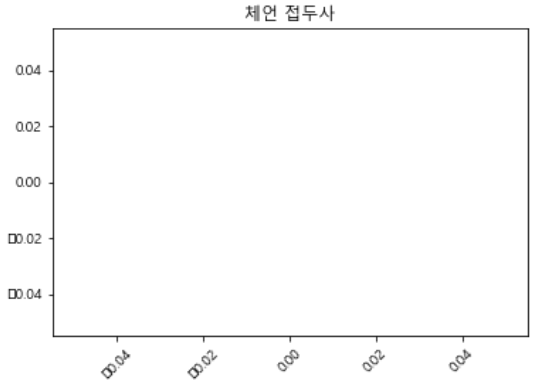

형용사
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00

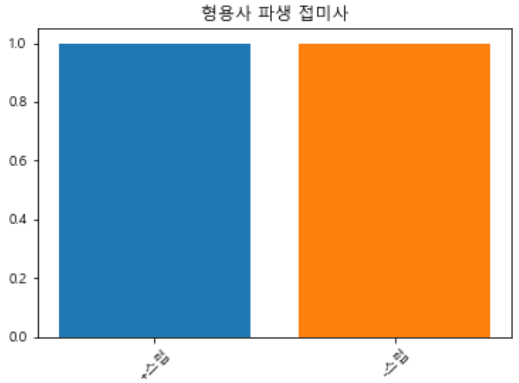

부정 지정사
1.0
0.8
0.6
0.4
0.2
0.0
+아니
-아니

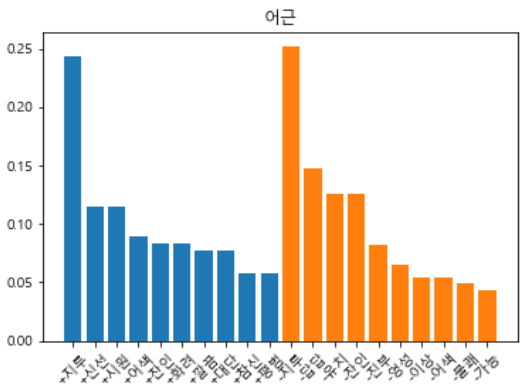

동사
0.25
0.20
0.15
0.10
0.05
0.00
+나오
+먼저
+보이
+오르
+느끼
+위하
+돌아오
+같
+죽이
+끝나
-나오
-먼저
-오르
-보이
-죽이
-끝나
-싸우
-열어지
-남기
-돌아오

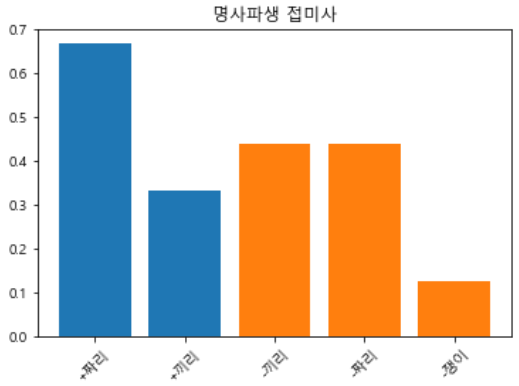

긍정 지정사
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04

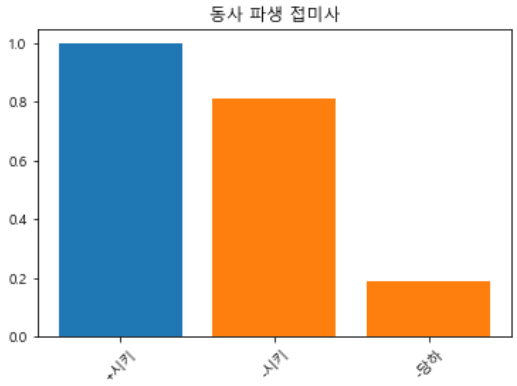

보조 용언
0.5
0.4
0.3
0.2
0.1
0.0
+버리
+못하
+나가
+가지
-못하
-버리
-드리
-나가
-빠지
-가지


체언 접두사
0.04
0.02
0.00
0.02
0.04
0.04
0.02
0.00
0.02
0.04


형용사 파생 접미사
1.0
0.8
0.6
0.4
0.2
0.0
+스럽
-스럽


어근
0.25
0.20
0.15
0.10
0.05
0.00


명사파생 접미사
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
+꾸러
+꺼리
-꺼리
-꾸러
-쟁이



In [29]:

```
from math import log
N = posN + negN
posY = posN / N
negY = negN / N
posPMI = defaultdict(float)
negPMI = defaultdict(float)
posNPMI = defaultdict(float)
negNPMI = defaultdict(float)

for t in list(set(posVoca + negVoca)):
    posX = (len(posTDM[t]) + len(negTDM[t])) / N

    posXY = (len(posTDM[t]) + 0.1) / N
    negXY = (len(negTDM[t]) + 0.1) / N

    posPMI[t] = log(posXY / (posX * negY))
    negPMI[t] = log(negXY / (posX * negY))

    posNPMI[t] = posPMI[t] / -log(posXY)
    negNPMI[t] = negPMI[t] / -log(negXY)
```

In [32]:

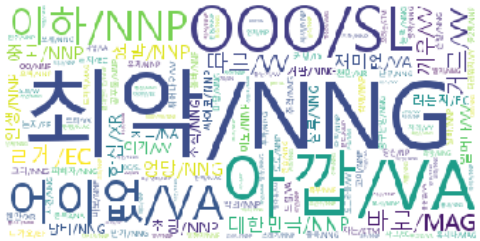
```
posSO = defaultdict(float)
negSO = defaultdict(float)
posNSO = defaultdict(float)
negNSO = defaultdict(float)

for t in list(set(posVoca + negVoca)):
    posSO[t] = posPMI[t] - negPMI[t]
    negSO[t] = negPMI[t] - posPMI[t]
    posNSO[t] = posNPMI[t] - negNPMI[t]
    negNSO[t]= negNPMI[t] - posNPMI[t]
```

In [34]:

```
from wordcloud import WordCloud
wc = WordCloud(font_path='C:/Windows/Fonts/malgun.ttf', background_color='white')

wc.generate_from_frequencies({'/'.join(t):so for t, so in posSO.items()})
plt.imshow(wc.to_array())
plt.axis('off')
plt.show()

wc.generate_from_frequencies({'/'.join(t):so for t, so in negSO.items()})
plt.imshow(wc.to_array())
plt.axis('off')
plt.show()
```

In [37]:

```
wc.generate_from_frequencies({'/'.join(t):nso for t, nso in posNSO.items()})
plt.imshow(wc.to_array())
plt.axis('off')
plt.show()

wc.generate_from_frequencies({'/'.join(t):nso for t, nso in negNSO.items()})
plt.imshow(wc.to_array())
plt.axis('off')
plt.show()
```



In [35]:

```
test = [
    "크리스찬 베일의 팬이라 봤는데 연기는 진짜 완벽했다.딕체니만 연구하고 파고든 이유를 충분히 보여줬다.911 이후 우리가 알지 못한 미국의 정치적 이야기가 흥미로웠다.이라크와 아프간 문제를 두고 대립하는 관료들의 설전은 실제로 보는 거처럼 리얼했다.",
    "영화보고 평점달긴 또 처음이네요…왜 이렇게 눈물이 나는지 모르겠네요ㅠ줄거리보다 마음이 느껴지는 한편의 인생을 보는것 같아요. 강추!!!",
    "소리 내 울지 않았던 아버지가 펑펑 우는 그 순간, 영화의 절정이라는 것을 깨달았다. 감동이 휘몰아 치면서 영화에 동화되었다. 하지만 중간에 등장했던 떡밥들이 어중간하게 풀리면서 찜찜함을 남겨주기도 한다.",
    "말이 필요 없네요 기다린 만큼 재미나게 보고 왔습니다",
    "초중반은 지루하고 마지막이 그나마 볼만함마블다른영화보다는 구성이 탄탄하지 않음",
    "영화라고하기엔;;드라마 스페셜 수준",
    "왜 하필 이런 3류영화는 영화 소개부터 꼴통형사로 시작하냐 ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ",
    "ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ 진짜 제목부터 보고싶지않은 영화는 처음이닼ㅋㅋㅋ"
]
```

In [36]:

```
for review in test:
    sentiment = 0.0
    termList = list()
    for t in ma.pos(review):
        if len(t[0]) > 1:
            termList.append(t)
    for t in termList:
        sentiment += posSO[t]
    if sentiment > 0.1:
        print('Positive')
    elif sentiment < -0.1:
        print('Negative')
    else:
        print('Neutral')
    print(sentiment)
    print(review)
```

```
Positive
8.467726424097537
크리스찬 베일의 팬이라 봤는데 연기는 진짜 완벽했다.딕체니만 연구하고 파고든 이유를 충분히 보여줬다.911 이후 우리가 알지 못한 미국의 정치적 이야기가 흥미로웠다.이라크와 아프간 문제를 두고 대립하는 관료들의 설전은 실제로 보는 거처럼 리얼했다.
Positive
0.7668306917508783
영화보고 평점달긴 또 처음이네요…왜 이렇게 눈물이 나는지 모르겠네요ㅠ줄거리보다 마음이 느껴지는 한편의 인생을 보는것 같아요. 강추!!!
Positive
9.912529442746402
소리 내 울지 않았던 아버지가 펑펑 우는 그 순간, 영화의 절정이라는 것을 깨달았다. 감동이 휘몰아 치면서 영화에 동화되었다. 하지만 중간에 등장했던 떡밥들이 어중간하게 풀리면서 찜찜함을 남겨주기도 한다.
Positive
5.352357260071789
말이 필요 없네요 기다린 만큼 재미나게 보고 왔습니다
Positive
3.880831588652648
초중반은 지루하고 마지막이 그나마 볼만함마블다른영화보다는 구성이 탄탄하지 않음
Negative
-2.511724599683303
영화라고하기엔;;드라마 스페셜 수준
Negative
-6.484189227131802
왜 하필 이런 3류영화는 영화 소개부터 꼴통형사로 시작하냐 ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ
Negative
-1.6757170820560772
ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ 진짜 제목부터 보고싶지않은 영화는 처음이닼ㅋㅋㅋ
```

In [38]:

```
for review in test:
    sentiment = 0.0
    termList = list()
    for t in ma.pos(review):
        if len(t[0]) > 1:
            termList.append(t)
    for t in termList:
        sentiment += posNSO[t]
    if sentiment > 0.1:
        print('Positive')
    elif sentiment < -0.1:
        print('Negative')
    else:
        print('Neutral')
    print(sentiment)
    print(review)
```

```
Positive
0.8709039166153996
크리스찬 베일의 팬이라 봤는데 연기는 진짜 완벽했다.딕체니만 연구하고 파고든 이유를 충분히 보여줬다.911 이후 우리가 알지 못한 미국의 정치적 이야기가 흥미로웠다.이라크와 아프간 문제를 두고 대립하는 관료들의 설전은 실제로 보는 거처럼 리얼했다.
Negative
-0.13786736917751655
영화보고 평점달긴 또 처음이네요…왜 이렇게 눈물이 나는지 모르겠네요ㅠ줄거리보다 마음이 느껴지는 한편의 인생을 보는것 같아요. 강추!!!
Positive
0.9379950473054057
소리 내 울지 않았던 아버지가 펑펑 우는 그 순간, 영화의 절정이라는 것을 깨달았다. 감동이 휘몰아 치면서 영화에 동화되었다. 하지만 중간에 등장했던 떡밥들이 어중간하게 풀리면서 찜찜함을 남겨주기도 한다.
Positive
0.7839282489630992
말이 필요 없네요 기다린 만큼 재미나게 보고 왔습니다
Positive
0.6365937266360837
초중반은 지루하고 마지막이 그나마 볼만함마블다른영화보다는 구성이 탄탄하지 않음
Negative
-0.5340835154902497
영화라고하기엔;;드라마 스페셜 수준
Negative
-0.9332419143025249
왜 하필 이런 3류영화는 영화 소개부터 꼴통형사로 시작하냐 ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ
Negative
-0.47270030159269566
ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ 진짜 제목부터 보고싶지않은 영화는 처음이닼ㅋㅋㅋ
```

In [41]:

```
N = len(reviews['리뷰'])
pSeed = ['예쁘', '유쾌']
nSeed = ['지겹', '한심']
```

In [42]:

```
DTM = defaultdict(Counter)
TDM = defaultdict(Counter)

for i,review in enumerate(reviews['리뷰']):
    for t in ma.pos(review):
        if len(t[0]) > 1:
            DTM[i][t[0]] += 1

for i,termList in DTM.items():
    for t,freq in termList.items():
        TDM[t][i] = freq
```

In [43]:

```
pSeedDocList = list(TDM[pSeed[0]].keys())
nSeedDocList = list(TDM[nSeed[0]].keys())

posY = len(pSeedDocList) / N
negY = len(nSeedDocList) / N

posPMI = defaultdict(float)
negPMI = defaultdict(float)
posNPMI = defaultdict(float)
negNPMI = defaultdict(float)

for t in TDM:
    xDocList = list(TDM[t].keys())
    posX = len(xDocList) / N

    xpDocList = set(pSeedDocList).intersection(xDocList)
    xnDocList = set(nSeedDocList).intersection(xDocList)

    posXY = (len(xpDocList) + 0.1) / N
    negXY = (len(xnDocList) + 0.1) / N

    posPMI[t] = log(posXY / (posX * negY))
    negPMI[t] = log(negXY / (posX * negY))

    posNPMI[t] = posPMI[t] / -log(posXY)
    negNPMI[t] = negPMI[t] / -log(negXY)
```

In [44]:

```
posSO = defaultdict(float)
negSO = defaultdict(float)
posNSO = defaultdict(float)
negNSO = defaultdict(float)

for t in TDM:
    posSO[t] = posPMI[t] - negPMI[t]
    negSO[t] = negPMI[t] - posPMI[t]
    posNSO[t] = posNPMI[t] - negNPMI[t]
    negNSO[t] = negNPMI[t] - posNPMI[t]
```

In [45]:

```
wc.generate_from_frequencies({''.join(t):so for t, so in posNSO.items()})
plt.imshow(wc.to_array())
plt.axis('off')
plt.show()

wc.generate_from_frequencies({''.join(t):so for t, so in negNSO.items()})
plt.imshow(wc.to_array())
plt.axis('off')
plt.show()
```

Another Sentiment Analysis

In [47]:

```
from konlpy.tag import Komoran

ma = Komoran()
documents = list()

for review in reviews["리뷰"]:
    documents.append([pos[0] for pos in ma.pos(review) if len(pos[0]) > 1 and pos[1] in ["NNG",
"NNP"]])
```

In [50]:

```
from random import randrange

a = 0.1
b = 0.1
K = 5

docTermTopicMat = list()
Vocabulary = list()

for doc in documents:
    termTopic = list()

    for term in doc:
        termTopic.append([term, randrange(K)])
        Vocabulary.append(term)

    docTermTopicMat.append(termTopic)

Vocabulary = list(set(Vocabulary))

M = len(docTermTopicMat)
N = len(Vocabulary)
```

In [48]:

```
TDM = defaultdict(lambda: defaultdict(int))

for docName, termList in enumerate(documents):
    for term in termList:
        TDM[term][docName] += 1
```

In [51]:

```
from math import log

pSeeds = ["감동", "최고", "행복"]
nSeeds = ["쓰레기", "억지", "최악"]

N = len(documents)
smoothing = 0.1

positivePMI = defaultdict(lambda: defaultdict(float))
positiveNPMI = defaultdict(lambda: defaultdict(float))

negativePMI = defaultdict(lambda: defaultdict(float))
negativeNPMI = defaultdict(lambda: defaultdict(float))

for seed in pSeeds:
    seedDocList = set(list(TDM[seed].keys()))
    seedLikelihood = len(seedDocList)/N

    for term in Vocabulary:
        termDocList = list(TDM[term].keys())
        termLikelihood = len(termDocList)/N

        jointLikelihood = (len(seedDocList.intersection(termDocList)) + smoothing)/N
        multiLikelihood = seedLikelihood * termLikelihood

        positivePMI[term][seed] = log(jointLikelihood/multiLikelihood)
        positiveNPMI[term][seed] = positivePMI[term][seed]/-log(jointLikelihood)


for seed in nSeeds:
    seedDocList = set(list(TDM[seed].keys()))
    seedLikelihood = len(seedDocList)/N

    for term in Vocabulary:
        termDocList = list(TDM[term].keys())
        termLikelihood = len(termDocList)/N

        jointLikelihood = (len(seedDocList.intersection(termDocList)) + smoothing)/N
        multiLikelihood = seedLikelihood * termLikelihood

        negativePMI[term][seed] = log(jointLikelihood/multiLikelihood)
        negativeNPMI[term][seed] = negativePMI[term][seed]/-log(jointLikelihood)
```

In [52]:

```
pmiSO = defaultdict(float)
npmiSO = defaultdict(float)

for term in Vocabulary:
    positivePmiSum = 0.0
    positiveNpmiSum = 0.0

    negativePmiSum = 0.0
    negativeNpmiSum = 0.0

    for seed in pSeeds:
        positivePmiSum += positivePMI[term][seed]
        positiveNpmiSum += positiveNPMI[term][seed]

    for seed in nSeeds:
        negativePmiSum += negativePMI[term][seed]
        negativeNpmiSum += negativeNPMI[term][seed]

    pmiSO[term] = positivePmiSum - negativePmiSum
    npmiSO[term] = positiveNpmiSum - negativeNpmiSum
```

In [53]:

```
positivePmiList = dict(sorted(pmiSO.items(), key=lambda x:x[1], reverse=True)[:30])
negativePmiList = dict(sorted(pmiSO.items(), key=lambda x:x[1], reverse=False)[:30])

positiveNpmiList = dict(sorted(npmiSO.items(), key=lambda x:x[1], reverse=True)[:30])
negativeNpmiList = dict(sorted(npmiSO.items(), key=lambda x:x[1], reverse=False)[:30])
```

In [56]:

```
from collections import defaultdict

topicTermMatrix = defaultdict(lambda:defaultdict(int))
docTopicMatrix = defaultdict(lambda:defaultdict(int))
topicCount = defaultdict(int)

for i, termTopic in enumerate(docTermTopicMat):
    for row in termTopic:
        topicTermMatrix[row[1]][row[0]] += 1
        docTopicMatrix[i][row[1]] += 1
        topicCount[row[1]] += 1
```

In [54]:

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

wc = WordCloud(font_path="C:/Windows/Fonts/malgun.ttf", background_color="white")

for termList, name in zip([positivePmiList, negativePmiList, positiveNpmiList, negativeNpmiList
], ["긍정-PMI", "부정-PMI", "긍정-NPMI", "부정-NPMI"]):
    print(name)
    print([term[0] for term in sorted(termList.items(), key=lambda x:x[1], reverse=True)])
    wc.generate_from_frequencies(termList)
    plt.imshow(wc.to_array())
    plt.rcParams["figure.figsize"] = (7,5)
    plt.axis("off")
    plt.show()
```

긍정-PMI
['행복', '알라딘', '스미스', '!!!', '지니', '최고', '마음', '진짜', '사랑', '후회', '더빙', '자스민', '성인', '가족', '디즈니', '사실', '감사', '감정', '아들', '원작', '엄마', '덕분', '명작', '완벽', '감동', '노래', '시대', '아이들', '자막', '장난감']

부정-PMI
['마동석', '오버', '김규리', '최악', '별점', '연출', '억지', '전개', '살인', '전개도', '악마', '조작', '깡패', '삼류', '동전', '유치', '커버', '컨셉', '복선', '인간', '동양', '짜깁기', '설정', '형사', '귀신', '경찰', '공짜', '쓰레기', '감독', '능력']

긍정-NPMI
['행복', '최고', '감동', '알라딘', '스미스', '!!!', '지니', '마음', '진짜', '사랑', '성인', '후회', '더빙', '자스민', '디즈니', '덕분', '노래', '사실', '비주얼', '우디', '명작', '가족', '아들', '감사', '토이', '감정', '자막', '완벽', '엄마', '장난감']

부정-NPMI
['납치', '오버', '마동석', '인간', '살인', '깡패', '개연', '전개도', '유치', '악마', '컨셉', '연출', '조작', '커버', '동전', '삼류', '동양', '경찰', '귀신', '형사', '복선', '짜깁기', '전개', '공짜', '감독', '설정', '능력', '최악', '억지', '쓰레기']

In [57]:

```
from matplotlib import font_manager

font = font_manager.FontProperties(fname="C:/Windows/Fonts/malgun.ttf").get_name()
plt.rcParams["font.family"] = font
plt.rcParams['axes.unicode_minus'] = False

print("{0:^8s} {1:^12s} {2:^6}".format("Topic", "PMI", "NPMI"))

for topic, title in zip(sorted(topicTermMatrix.keys()), ["알라딘", "토이스토리", "공포", "드라마", "액션"]):
    sentimentPMI = 0.0
    sentimentNPMI = 0.0

    topicWords = sorted(topicTermMatrix[topic].items(), key=lambda x:x[1], reverse=True)[:40]

    for term in topicWords:
        sentimentPMI += pmiSO[term[0]]
        sentimentNPMI += npmiSO[term[0]]

    print("{0:ㅇ>5s} {1:>+10.4f} {2:>+9.4f}".format(title, sentimentPMI, sentimentNPMI))
    plt.barh(title, sentimentNPMI)
plt.show()
```

Topic	PMI	NPMI
ㅇㅇ알라딘	+5.5685	+1.4898
토이스토리	-18.3577	-1.1688
ㅇㅇㅇ공포	-22.5739	-1.8197
ㅇㅇ드라마	-26.1040	-1.9518
ㅇㅇㅇ액션	+3.7960	+1.4269