In [78]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [80]:
```
ri = pd.read_csv('cases_ri.csv')
```

In [81]:
```
ri.shape
```

Out[81]:
```
(239, 5)
```

In [82]:
```
ri.head()
```

Out[82]:

	file name	title	type	decision	text
0	06-290.pdf	State v. Michael Tetreault, No. 06-290 (June 1...	criminal	affirmed	['', 'Supreme Court', 'No. 2006-290-C.A.', '(P...
1	08-27.pdf	State v. Thomas P. Byrne, No. 08-27 (June 19, ...	criminal	not affirmed	['', 'Supreme Court', 'No. 2008-27-C.A.', '(P2...
2	07-108.pdf	State v. Robert Collazo, No. 07-108 (April 3, ...	criminal	affirmed	['', 'Supreme Court', 'No. 2007-108-C.A.', '(P...
3	07-334.pdf	State v. Samuel Adewumi, No. 07-334 (March 17,...	criminal	affirmed	['', 'Supreme Court', 'No. 2007-334-C.A.', '(W...
4	07-123.pdf	State v. Phillip Jackson, No. 07-123 (March 20...	criminal	not affirmed	['', 'Supreme Court', 'No. 2007-123-C.A.', '(P...

In [83]:
```
ri.type.unique()
```

Out[83]:
```
array(['criminal'], dtype=object)
```

In [84]:
```
criminal = ri[ri['type']=='criminal']   # there are 269 criminal cases in N
criminal.shape
```

Out[84]:
```
(239, 5)
```

In [ ]:
```

```

In [85]:
```
criminal.decision.unique()
```

Out[85]:
```
array(['affirmed', 'not affirmed', 'affirm in part'], dtype=object)
```

In [86]:
```
criminal.decision = criminal.decision.str.lower().copy()
part =  criminal[criminal.decision.str.contains('part')]
part.shape
```

Out[86]:
```
(5, 5)
```

In [87]:
```
no_part = criminal[~criminal.decision.str.contains('part')]
no_part.shape
```

Out[87]: (234, 5)

In [88]:
```
reversed = no_part[no_part.decision.str.contains('not')]
reversed.shape
```

Out[88]: (28, 5)

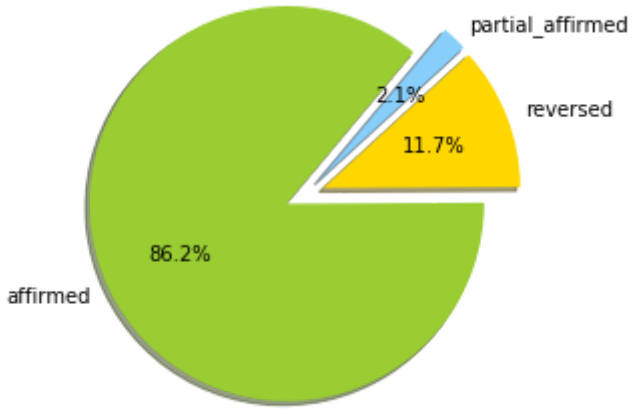
In [89]:
```
affirmed = no_part[~no_part.decision.str.contains('not')]
affirmed.shape
```

Out[89]: (206, 5)

In [90]:
```
#affirmed = criminal[criminal.decision=='Affirmed.']   # separate the affir
#reversed = criminal[criminal.decision=='Reversed.']
#partial_reversed = criminal[criminal.decision!='Affirmed.'][criminal.decis
```

The proportion of affirmed cases is 71.4%, the proportion of reversed cases is 3.3%, the proportion of partial reversed cases is 25.3.

In [91]:
```
labels='affirmed','reversed','partial_affirmed'
sizes= (len(affirmed)/len(criminal))*100,(len(reversed)/len(criminal))*100,
colors='yellowgreen','gold','lightskyblue'
explode=0.1,0.1,0.1
plt.pie(x=sizes,explode=explode,labels=labels,colors=colors,autopct='%1.1f%
plt.axis('equal')
plt.show()
```



In [92]:
```
informant = criminal[criminal.text.str.contains('informant')]
non_informant =  criminal[~criminal.text.str.contains('informant')]
```

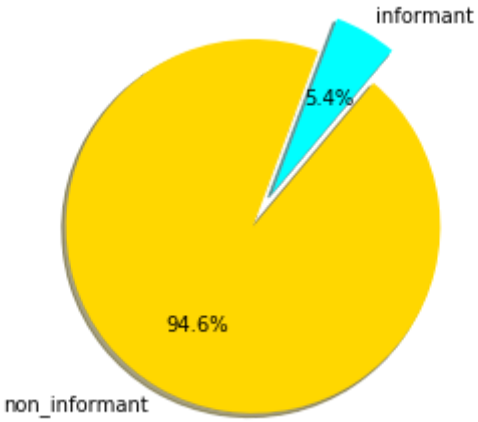
In [93]:
```
informant.shape
```

Out[93]:
```
(13, 5)
```

In [ ]:
```

```

In [94]:
```
labels='informant','non_informant'
sizes= (len(informant)/len(criminal))*100,(len(non_informant)/len(criminal)
colors='aqua','gold'
explode=0.1,0.1
plt.pie(x=sizes,explode=explode,labels=labels,colors=colors,autopct='%1.1f%
plt.axis('equal')
plt.show()
```



There are some common features in affirmed and reversed, like 'case','court','new hampshire', this words appear many times, but it doesn't make senses. So we delete these words. Meanwhile, there are many numbers in the case. The numbers usually appear in two places: the case number and the law number. We can't tell where the numbers come from. So we delete the number.

In [95]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [96]:
```
def tfidf(X,n):
    vectorizer = TfidfVectorizer(stop_words='english',max_features=n)
    tfidf = vectorizer.fit_transform(X)
    word = vectorizer.get_feature_names()
    return word
```

In [97]:
```
criminal_common_word = tfidf(ri['text'],20)
criminal_common_word +=['rhode','island','may','two','due','also','iii','ev
criminal_common_word
```

Out[97]:
```
['2d',
 'appeal',
 'case',
 'counsel',
 'court',
 'defendant',
 'did',
 'evidence',
 'hearing',
 'jury',
 'justice',
 'motion',
 'mr',
 'opinion',
 'police',
 'rule',
 'state',
 'testified',
 'testimony',
 'trial',
 'rhode',
 'island',
 'may',
 'two',
 'due',
 'also',
 'iii',
 'even',
 'though',
 'whether']
```

In [98]:
```
# import re
import re
def clean_common_word(text):            # delect the number, punctuation and
    text =  re.sub("[^a-zA-Z#]", " ", text)
    words=text.lower().split()           # lower case
    words = [w for w in words if len(w)>=3]
    stoplist = stopwords.words('english')
    words = [word for word in words if word not in stoplist]
    words = [word for word in words if word not in criminal_common_word]
    return " ".join(words)
```

analyze the affirmed cases and reversed cases

In [99]:
```
def handle_all_cases(df):
    for i in range(len(df)):
        df.text.iloc[i] = clean_common_word(df.text.iloc[i])
    return df
```

In [106]:
```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/wangqitong/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Out[106]:
```
True
```

In [107]:
```
reversed = handle_all_cases(reversed)
affirmed = handle_all_cases(affirmed)
criminal = handle_all_cases(criminal)
part = handle_all_cases(part)
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-pack
ages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  This is separate from the ipykernel package so we can avoid doing impor
ts until
```

In [108]:
```
reversed_word = tfidf(reversed['text'],10)     # select the most important
reversed_word
```

Out[108]:
```
['criminal',
 'defense',
 'fact',
 'law',
 'officer',
 'review',
 'right',
 'superior',
 'time',
 'witness']
```

In [109]:
```
affirmed_word = tfidf(affirmed['text'],10)
affirmed_word
```

Out[109]:
```
['criminal',
 'defense',
 'new',
 'quoting',
 'stated',
 'statement',
 'superior',
 'supreme',
 'time',
 'witness']
```

In [110]:
```
criminal_word = tfidf(criminal['text'],10)
criminal_word
```

Out[110]:
```
['criminal',
 'defense',
 'new',
 'officer',
 'quoting',
 'stated',
 'superior',
 'supreme',
 'time',
 'witness']
```

In [111]:
```
part_word = tfidf(part['text'],10)
part_word
```

Out[111]:
```
['apartment',
 'cell',
 'conditions',
 'phone',
 'probation',
 'sentence',
 'stated',
 'text',
 'time',
 'violation']
```

When we select the most import 15 features in reversed cases and affirmed cases, we found reversed cases contains features: child and sexual.

In [ ]:
```

```

analyze the cases with informant and without

In [112]:
```
non_informant = handle_all_cases(non_informant)
informant = handle_all_cases(informant)
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-pack
ages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  This is separate from the ipykernel package so we can avoid doing impor
ts until
```

In [113]:
```
non_informant_word = tfidf(non_informant['text'],10)
non_informant_word
```

Out[113]:
```
['criminal',
 'defense',
 'new',
 'officer',
 'quoting',
 'stated',
 'superior',
 'supreme',
 'time',
 'witness']
```

In [114]:
```
informant_word = tfidf(informant['text'],10)
informant_word
```

Out[114]:
```
['ciresi',
 'criminal',
 'defendants',
 'detective',
 'information',
 'murder',
 'quoting',
 'review',
 'statement',
 'time']
```

reversed cases with informant

In [115]:
```
informant_in_reverse = reversed[reversed.text.str.contains('informant')]
non_informant_in_reverse =  reversed[~reversed.text.str.contains('informant
```

In [116]:
```
informant_in_reverse
```

Out[116]:

	file name	title	type	decision	text
21	07-30.pdf	State v. James Oliveira, No. 07-0030 (December...	criminal	not affirmed	supreme james oliveira notice subject formal r...
70	08-53.pdf	State v. Robinson Berroa, No. 08-53 (November ...	criminal	not affirmed	supreme robinson berroa notice subject formal ...
210	13-124.pdf	State v. Victor Arciliares, No. 13-124 (Januar...	criminal	not affirmed	supreme victor arciliares notice subject forma...

In [117]:
```
non_info_r_word = tfidf(non_informant_in_reverse['text'],10)
non_info_r_word
```

Out[117]:
```
['criminal',
 'defense',
 'fact',
 'issue',
 'law',
 'officer',
 'review',
 'superior',
 'time',
 'witness']
```

In [118]:
```
info_r_word = tfidf(informant_in_reverse['text'],10)
info_r_word
```

Out[118]:
```
['baccaire',
 'child',
 'defense',
 'det',
 'error',
 'information',
 'laforest',
 'phillip',
 'right',
 'said']
```

The reversed cases with informant is not related to child and sexual.

affirmed cases with informant

In [119]:
```
informant_in_affirmed = affirmed[affirmed.text.str.contains('informant')]
non_informant_in_affirmed =  affirmed[~affirmed.text.str.contains('informan
```

In [120]:
```
info_a_word = tfidf(informant_in_affirmed['text'],10)
info_a_word
```

Out[120]:
```
['ciresi',
 'criminal',
 'defendants',
 'degree',
 'detective',
 'murder',
 'quoting',
 'review',
 'statement',
 'warrant']
```

In [121]:
```
non_info_a_word = tfidf(non_informant_in_affirmed['text'],10)
non_info_a_word
```

Out[121]:
```
['assault',
 'defense',
 'new',
 'quoting',
 'stated',
 'statement',
 'superior',
 'supreme',
 'time',
 'witness']
```

In [ ]:

In affirmed cases with informant, the feature 'murder' may be important

In [ ]:

similarity

In [122]:
```
a = []
for i in range(len(reversed)):
    a.append(reversed.text.iloc[i].split(' '))
```

In [123]:
```
#a = reversed.text.iloc[24].split(' ')
```

In [124]:
```
from gensim.models import Word2Vec
```

In [125]:
```
word2vec_model =Word2Vec(a,min_count=10)
```

In [126]:
```
print(word2vec_model)
```

```
Word2Vec(vocab=1775, size=100, alpha=0.025)
```

In [133]:
```
word2vec_model.wv.most_similar('reversed')
```

Out[133]:
```
[('remand', 0.9991968274116516),
 ('district', 0.9991153478622437),
 ('instant', 0.99909508228302),
 ('specifically', 0.999093770980835),
 ('new', 0.9990875720977783),
 ('pepper', 0.9990874528884888),
 ('conclusion', 0.9990872144699097),
 ('shower', 0.9990850687026978),
 ('including', 0.9990758895874023),
 ('close', 0.9990744590759277)]
```

In [134]:
```
reversed.iloc[0]
```

Out[134]:
```
file name                                                08-27.pdf
title        State v. Thomas P. Byrne, No. 08-27 (June 19, ...
type                                                    criminal
decision                                            not affirmed
text         supreme thomas byrne present goldberg acting f...
Name: 1, dtype: object
```

In [ ]:
```

```

In [141]:
```
b=reversed.text.iloc[2].split(' ')
```

In [142]:
```
word2vec_model1 =Word2Vec([b],min_count=1)
```

In [143]:
```
word2vec_model1.wv.most_similar('reversed')
```

Out[143]:
```
[('fact', 0.2718867063522339),
 ('legislature', 0.27143365144729614),
 ('waived', 0.24244937300682068),
 ('communities', 0.23680533468723297),
 ('test', 0.23623059689998627),
 ('emphasis', 0.22708189487457275),
 ('restricted', 0.22419720888137817),
 ('order', 0.22389544546604156),
 ('assert', 0.21679246425628662),
 ('overlook', 0.2111416608095169)]
```

In [ ]:
```

```