In [233…
```
import pandas as pd
import numpy as np
from numpy.random import seed

import warnings
warnings.filterwarnings("ignore")
```

In [234…
```
seed(100)
```

In [235…
```
df = pd.read_csv("./DATA/critics.csv")
df.shape
```

Out[235… (27631, 8)

In [236…
```
n = len(df)
n1 = 7000
x = list(np.arange(n))
idx = list(np.random.choice(x,n1))
```

In [237…
```
from collections import Counter
```

In [238…
```
c = Counter(idx)
type(c)
```

Out[238… collections.Counter

In [239…
```
c.most_common()[0:5]
```

Out[239… [(22788, 4), (26223, 4), (16273, 4), (18552, 4), (14821, 4)]

In [240…
```
c.most_common()[0]
```

Out[240… (22788, 4)

In [241…
```
c1 = list(c)
c1[0:5]
```

Out[241… [5640, 23320, 14147, 24423, 12119]

In [242…
```
np.shape(c1)
```

Out[242… (6210,)

In [243…
```
clu = np.unique(c1)
np.shape(clu)
```

Out[243… (6210,)

In [ ]:

In [244…
```
df = df.loc[c1, :]
df.shape
```

Out[244… (6210, 8)

In [245…
```
df.head()
```

Out[245…

	critic	fresh	imdb	publication	quote	review_date	rtid	title
5640	James Berardinelli	fresh	107225	ReelViews	NaN	2000-01-01	13478	lo speriamo che me la cavo
23320	Roger Ebert	fresh	107798	Chicago Sun-Times	A clever device to take your mind off your pro…	2000-01-01	10452	The Pelican Brief
14147	NaN	none	118901	Washington Post	NaN	2007-08-18	15672	Critical Care
24423	David Jenkins	fresh	77405	Time Out	Visually and thematically, it's still one of t…	2011-08-31	10859	Days of Heaven
12119	James Berardinelli	rotten	118750	ReelViews	NaN	2000-01-01	15333	Booty Call

In [246…
```
cols = df.columns.tolist()
cols
```

Out[246…
```
['critic',
 'fresh',
 'imdb',
 'publication',
 'quote',
 'review_date',
 'rtid',
 'title']
```

In [247…
```
cols1 = ['quote', 'fresh']
df = df[cols1]
df.shape
```

Out[247… (6210, 2)

In [ ]:

In [248…
```
df.fresh.value_counts()
```

Out[248…
```
fresh     2765
rotten    1836
none      1609
Name: fresh, dtype: int64
```

In [249…
```
df = df[ ~ (df["fresh"]=="none") ]
df.shape
```

Out[249… (4601, 2)

from 27.6k -> 20.3k

In [250…
```
df.quote.isnull().value_counts()
```

Out[250…
```
False    3532
True     1069
Name: quote, dtype: int64
```

In [251…
```
df = df[ df.quote.isnull()==False]
df.shape
```

Out[251… (3532, 2)

From 20.3k to 15.5k

In [252…
```
print(" Almost ", str(np.round((27631-15534)*100/27631),0) ) , "% of the data is wrangled out")
```

```
 Almost  44.0 % of the data is wrangled out
```

In [ ]:

Feature Extraction

In [253…
```
from sklearn.feature_extraction.text import CountVectorizer
vec = CountVectorizer()
```

In [254…
```
X = vec.fit_transform(df.quote.values).tocsc()
X.shape, type(X)
```

Out[254… ((3532, 10868), scipy.sparse.csc.csc_matrix)

In [255…
```
df.fresh.value_counts()
```

Out[255…
```
fresh     2216
rotten    1316
Name: fresh, dtype: int64
```

In [256…
```
yv = df.fresh.value_counts().index.tolist()
yv
```

Out[256… ['fresh', 'rotten']

In [257…
```
df.replace(to_replace=yv, value=[1,0], inplace=True)
df['fresh'].value_counts()
```

Out[257…
```
1    2216
0    1316
Name: fresh, dtype: int64
```

In [258…
```
y = df.fresh.values
y[0:40]
```

Out[258…
```
array([1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0,
       0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0])
```

In [259…
```
X.shape, y.shape
```

Out[259… ((3532, 10868), (3532,))

Machine Learning

In [260…
```
from sklearn.model_selection import train_test_split, GridSearchCV
```

In [261…
```
x_tr, x_t, y_tr, y_t = train_test_split(X, y, test_size=0.3, random_state=100)
x_tr.shape, x_t.shape, y_tr.shape, y_t.shape
```

Out[261… ((2472, 10868), (1060, 10868), (2472,), (1060,))

0.765071872988629 with MultinomialNB

In [262…
```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, roc_auc_score
```

In [263…
```
clf = MultinomialNB()
clf.fit(x_tr, y_tr)
```

Out[263… MultinomialNB()

In [264…
```
y_p = clf.predict(x_t)
accuracy_score(y_t, y_p)
```

Out[264… 0.7264150943396226

In [ ]:

In [265…
```
from sklearn import svm
from sklearn.svm import SVC
```

In [266…
```
clf = GridSearchCV(svm.SVC(gamma='auto'), {'C':[1,10, 20], 'kernel':['linear']}, cv=5, return_train_score=False)
```

In [267…
```
clf.fit(x_tr, y_tr)
res = pd.DataFrame(clf.cv_results_)
```

In [268…
```
res.columns
```

Out[268…
```
Index(['mean_fit_time', 'std_fit_time', 'mean_score_time', 'std_score_time',
       'param_C', 'param_kernel', 'params', 'split0_test_score',
       'split1_test_score', 'split2_test_score', 'split3_test_score',
       'split4_test_score', 'mean_test_score', 'std_test_score',
       'rank_test_score'],
      dtype='object')
```

In [269…
```
res[ ['param_C', 'param_kernel', 'mean_test_score'] ]
```

Out[269…

	param_C	param_kernel	mean_test_score
0	1	linear	0.691355
1	10	linear	0.692168
2	20	linear	0.692168

0.704041

In [ ]: