In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df = pd.read_csv('../TextFiles/moviereviews.tsv', sep = '\t')
df.head()
```

Out[2]:

	label	review
0	neg	how do films like mouse hunt get into theatres...
1	neg	some talented actresses are blessed with a dem...
2	pos	this has been an extraordinary year for austra...
3	pos	according to hollywood movies made in last few...
4	neg	my first press screening of 1998 and already i...

In [3]:
```
len(df)
```

Out[3]: 2000

In [5]:
```
# Check a negative review of the movie -- first review
print(df['review'][0])
```

```
how do films like mouse hunt get into theatres ?
isn't there a law or something ?
this diabolical load of claptrap from steven speilberg's dreamworks st
udio is hollywood family fare at its deadly worst .
mouse hunt takes the bare threads of a plot and tries to prop it up wi
th overacting and flat-out stupid slapstick that makes comedies like j
ingle all the way look decent by comparison .
writer adam rifkin and director gore verbinski are the names chiefly r
esponsible for this swill .
the plot , for what its worth , concerns two brothers ( nathan lane an
d an appalling lee evens ) who inherit a poorly run string factory and
a seemingly worthless house from their eccentric father .
deciding to check out the long-abandoned house , they soon learn that
it's worth a fortune and set about selling it in auction to the highes
t bidder .
but battling them at every turn is a very smart mouse , happy with his
run-down little abode and wanting it to stay that way .
the story alternates between unfunny scenes of the brothers bickering
over what to do with their inheritance and endless action sequences as
the two take on their increasingly determined furry foe .
whatever promise the film starts with soon deteriorates into boring di
alogue , terrible overacting , and increasingly uninspired slapstick t
hat becomes all sound and fury , signifying nothing .
the script becomes so unspeakably bad that the best line poor lee even
s can utter after another run in with the rodent is : " i hate that mo
use " .
oh cringe !
this is home alone all over again , and ten times worse .
one touching scene early on is worth mentioning .
we follow the mouse through a maze of walls and pipes until he arrives
at his makeshift abode somewhere in a wall .
he jumps into a tiny bed , pulls up a makeshift sheet and snuggles up
to sleep , seemingly happy and just wanting to be left alone .
it's a magical little moment in an otherwise soulless film .
a message to speilberg : if you want dreamworks to be associated with
some kind of artistic credibility , then either give all concerned in
mouse hunt a swift kick up the arse or hire yourself some decent write
rs and directors .
this kind of rubbish will just not do at all .
```

In [6]:
```
# Check a positive review of the movie -- 3rd review
print(df['review'][2])
```

```
this has been an extraordinary year for australian films .
 " shine " has just scooped the pool at the australian film institute
```

```
awards , picking up best film , best actor , best director etc . to th
at we can add the gritty " life " ( the anguish , courage and friendsh
ip of a group of male prisoners in the hiv-positive section of a jail
) and " love and other catastrophes " ( a low budget gem about straigh
t and gay love on and near a university campus ) .
i can't recall a year in which such a rich and varied celluloid librar
y was unleashed from australia .
 " shine " was one bookend .
stand by for the other one : " dead heart " .
>from the opening credits the theme of division is established .
the cast credits have clear and distinct lines separating their first
and last names .
bryan | brown .
in a desert settlement , hundreds of kilometres from the nearest town
, there is an uneasy calm between the local aboriginals and the handfu
l of white settlers who live nearby .
the local police officer has the task of enforcing " white man's justi
ce " to the aboriginals .
these are people with a proud 40 , 000 year heritage behind them .
naturally , this includes their own system of justice ; key to which i
s " payback " .
an eye for an eye .
revenge .
usually extracted by the spearing through of the recipient's thigh .
brown , as the officer , manages quite well to keep the balance .
he admits that he has to 'bend the rules' a bit , including actively e
ncouraging at least one brutal " payback " .
 ( be warned that this scene , near the start , is not for the squeami
sh ) .
the local priest - an aboriginal , but in the " white fellas " church
- has a foot on either side of the line .
he is , figuratively and literally , in both camps .
ernie dingo brings a great deal of understanding to this role as the m
an in the middle .
he is part churchman and part politician .
however the tension , like the heat , flies and dust , is always there
.
whilst her husband - the local teacher - is in church , white lady kat
e ( milliken ) and her aborginal friend tony , ( pedersen ) have gone
off into the hills .
he takes her to a sacred site , even today strictly men-only .
she appears to not know this .
tony tells her that this is a special place , an initiation place .
he then makes love to her , surrounded by ancient rock art .
the community finds out about this sacrilegious act and it's payback t
ime .
the fuse is lit and the brittle inter-racial peace is shattered .
everyone is affected in the fall out .
to say more is to give away the details of this finely crafted film .
suffice to say it's a rewarding experience .
```

```
bryan brown , acting and co-producing , is the pivotal character .
his officer is real , human and therefore flawed .
brown comments that he expects audiences to feel warmth towards the ma
n , then suddenly feel angry about him .
it wasn't long ago that i visited central australia - ayers rock ( ulu
ru ) and alice springs - for the first time .
the wide-screen cinematography shows the dead heart of australia in a
way that captures it's vicious beauty , but never deteriorates into a
moving slide show , in which the gorgeous background dominates those p
esky actors in the foreground .
the cultural clash has provided the thesis for many a film ; from the
western to the birdcage .
at least three excellent australian films have covered the aboriginal
people and the line between them and we anglo-saxon 'invaders' : " jed
da " , " the chant of jimmie blacksmith " and " the last wave " .
in a year when the race 'debate' has reared up in australia , it is no
urishing to see such an intelligent , non-judgemental film as " dead h
eart " .
the aboriginal priest best sums this up .
he is asked to say if he is a " black fella or white fella " .
```

In [7]:
```
# Check for missing data
df.isnull().sum()
```

Out[7]:
```
label      0
review    35
dtype: int64
```

In [8]:
```
df.dropna(inplace = True)
```

In [9]:
```
df.isnull().sum()
```

Out[9]:
```
label     0
review    0
dtype: int64
```

In [11]:
```
#Check whether the review is an empty string filled with whitespace -- e
mystring = 'hello'
mystring.isspace()
```

Out[11]:
```
False
```

In [12]:
```
empty_string = '      '
empty_string.isspace()
```

Out[12]:
```
True
```

In [13]:
```
# Get the index position of blank reviews
blanks = []

for i, lb, rv in df.itertuples():
    if rv.isspace():
        blanks.append(i)
```

In [14]:
```
len(blanks)
```

Out[14]:
```
27
```

In [15]:
```
# Drop the blank reviews using their index positions in the list blanks
df.drop(blanks, inplace = True)
```

In [16]:
```
len(df)
```

Out[16]:
```
1938
```

In [18]:
```
df['label'].value_counts()
```

Out[18]:
```
pos    969
neg    969
Name: label, dtype: int64
```

Split the data into training and testing data

In [19]:
```
X = df['review']
y = df['label']
```

In [20]:
```
from sklearn.model_selection import train_test_split
```

In [21]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.3,
                                                    random_state=42)
```

In [22]:
```
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import LinearSVC
```

In [29]:
```
text_clf = Pipeline([('tfidf', TfidfVectorizer()), ('clf', LinearSVC())]
```

In [31]:
```
text_clf.fit(X_train, y_train)
```

Out[31]:
```
Pipeline(memory=None,
     steps=[('tfidf', TfidfVectorizer(analyzer='word', binary=False, d
ecode_error='strict',
        dtype=<class 'numpy.float64'>, encoding='utf-8', input='conten
t',
        lowercase=True, max_df=1.0, max_features=None, min_df=1,
        ngram_range=(1, 1), norm='l2', preprocessor=None, smooth_idf=T
rue,...ax_iter=1000,
     multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
     verbose=0))])
```

In [32]:
```
predictions = text_clf.predict(X_test)
```

In [33]:
```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, predictions))
```

```
[[235  47]
 [ 41 259]]
```

In [35]:
```
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

         neg       0.85      0.83      0.84       282
         pos       0.85      0.86      0.85       300

   micro avg       0.85      0.85      0.85       582
   macro avg       0.85      0.85      0.85       582
weighted avg       0.85      0.85      0.85       582
```

In [36]:
```
from sklearn import metrics
print(metrics.accuracy_score(y_test, predictions))
```

```
0.8487972508591065
```

In [ ]: