In [2]:
```
import pandas as pd
import numpy as np

import ast

import seaborn as sns
from matplotlib import pyplot
%matplotlib inline
```

In [10]:
```
import os
os.path
```

Out[10]:
```
<module 'ntpath' from 'C:\\ProgramData\\Anaconda3\\lib\\ntpath.py'>
```

import data and check data types

In [12]:
```
# movie = pd.read_csv('movies_metadata.csv',dtype={'popularity':'float64'})
movie = pd.read_csv('..\data\movies_metadata.csv',low_memory=False)
```

In [35]:
```
# get the column names
movie.columns
```

Out[35]:
```
Index(['adult', 'belongs_to_collection', 'budget', 'genres', 'homepage', 'i
d',
       'imdb_id', 'original_language', 'original_title', 'overview',
       'popularity', 'poster_path', 'production_companies',
       'production_countries', 'release_date', 'revenue', 'runtime',
       'spoken_languages', 'status', 'tagline', 'title', 'video',
       'vote_average', 'vote_count'],
      dtype='object')
```

In [36]:
```
# get an overview of the dataset and print the column dtypes
movie.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45466 entries, 0 to 45465
Data columns (total 24 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   adult                  45466 non-null  object
 1   belongs_to_collection  4494 non-null   object
 2   budget                 45466 non-null  object
 3   genres                 45466 non-null  object
 4   homepage               7782 non-null   object
 5   id                     45466 non-null  object
 6   imdb_id                45449 non-null  object
 7   original_language      45455 non-null  object
 8   original_title         45466 non-null  object
 9   overview               44512 non-null  object
 10  popularity             45461 non-null  object
 11  poster_path            45080 non-null  object
 12  production_companies   45463 non-null  object
 13  production_countries   45463 non-null  object
 14  release_date           45379 non-null  object
 15  revenue                45460 non-null  float64
 16  runtime                45203 non-null  float64
 17  spoken_languages       45460 non-null  object
 18  status                 45379 non-null  object
 19  tagline                20412 non-null  object
 20  title                  45460 non-null  object
 21  video                  45460 non-null  object
 22  vote_average           45460 non-null  float64
 23  vote_count             45460 non-null  float64
dtypes: float64(4), object(20)
memory usage: 8.3+ MB
```

In [37]:
```
print(movie.sample(3))
```

```
       adult belongs_to_collection   budget  \
34991  False                   NaN        0
15875  False                   NaN  1500000
41805  False                   NaN        0

                                              genres  \
34991                    [{'id': 37, 'name': 'Western'}]
15875  [{'id': 28, 'name': 'Action'}, {'id': 27, 'nam...
41805  [{'id': 28, 'name': 'Action'}, {'id': 99, 'nam...

                            homepage      id    imdb_id original_language  \
34991                            NaN   73740  tt0063479                it
15875  http://www.redcanyonmovie.com/   40599  tt1064801                en
41805                            NaN  142293  tt2723912                en

                     original_title  \
34991  Quella sporca storia nel west
15875                     Red Canyon
41805                Bending Colours

                                              overview  ... release_date  \
34991  On his way back from the Civil War, Johnny Ham...  ...   1968-03-22
15875  In Red Canyon, Regina and Devon return to thei...  ...   2008-01-01
41805  Bending Colours is a moving portrait of Jordy ...  ...   2012-01-01

      revenue runtime                          spoken_languages    status  \
34991     0.0    91.0  [{'iso_639_1': 'it', 'name': 'Italiano'}]  Released
15875     0.0    98.0   [{'iso_639_1': 'en', 'name': 'English'}]  Released
41805     0.0    38.0   [{'iso_639_1': 'en', 'name': 'English'}]  Released

       tagline            title  video vote_average vote_count
34991      NaN    Johnny Hamlet  False          7.3        6.0
15875      NaN       Red Canyon  False          4.8        4.0
41805      NaN  Bending Colours  False          4.3        2.0

[3 rows x 24 columns]
```

remove duplicate rows

In [38]:
```
# find duplicate data
duplicate = movie[movie.duplicated()]
duplicate
```

Out[38]:

	adult	belongs_to_collection	budget	genres	homepage
1465	False	NaN	0	[{'id': 18, 'name': 'Drama'}, {'id': 10749, 'n...	NaN
9165	False	NaN	0	[{'id': 80, 'name': 'Crime'}, {'id': 18, 'name...	NaN
9327	False	NaN	0	[{'id': 12, 'name': 'Adventure'}, {'id': 16, '...	NaN
13375	False	NaN	0	[{'id': 53, 'name': 'Thriller'}, {'id': 9648, ...	NaN
16764	False	NaN	0	[{'id': 53, 'name': 'Thriller'}, {'id': 9648, ...	NaN
21165	False	NaN	0	[{'id': 14, 'name': 'Fantasy'}, {'id': 18, 'na...	NaN
21854	False	NaN	0	[{'id': 18, 'name': 'Drama'}, {'id': 878, 'nam...	NaN
22151	False	NaN	0	[{'id': 28, 'name': 'Action'}, {'id': 27, 'nam...	http://www.daysofdarknessthemovie.com/
23044	False	NaN	0	[{'id': 18, 'name': 'Drama'}]	NaN
24844	False	NaN	0	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'nam...	http://www.dealthemovie.com/
28860	False	NaN	0	[{'id': 18, 'name': 'Drama'}, {'id': 35, 'name...	NaN
29374	False	NaN	0	[{'id': 18, 'name': 'Drama'}, {'id': 10769, 'n...	NaN
35798	False	{'id': 158365, 'name': 'Why We Fight', 'poster...	0	[{'id': 99, 'name': 'Documentary'}]	NaN
38871	False	NaN	0	[{'id': 28, 'name': 'Action'}, {'id': 18, 'nam...	NaN

	adult	belongs_to_collection	budget	genres	homepage
40040	False	NaN	980000	[{'id': 18, 'name': 'Drama'}, {'id': 14, 'name...	NaN
40276	False	NaN	0	[{'id': 35, 'name': 'Comedy'}]	NaN
45265	False	NaN	0	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'nam...	NaN

17 rows × 24 columns

In [39]:
```
# remove duplicate data
movie.drop(movie[movie.duplicated()].index,inplace=True)
```

clean irrelevant observations

In [40]:
```
movie.status.unique()
```

Out[40]:
```
array(['Released', nan, 'Rumored', 'Post Production', 'In Production',
       'Planned', 'Canceled'], dtype=object)
```

In [41]:
```
# filter only released movie
movie=movie[movie.status=='Released']
# mostly na columns and columns with similar information to other columns
movie=movie.drop(columns=['belongs_to_collection','budget','revenue', 'homepag
e', 'poster_path', 'spoken_languages','status','tagline','original_title'])
```

clean structure errors

In [42]:
```
# get an overview of the dataset and print the column dtypes
movie.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 44998 entries, 0 to 45465
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   adult                 44998 non-null  object
 1   genres                44998 non-null  object
 2   id                    44998 non-null  object
 3   imdb_id               44983 non-null  object
 4   original_language     44988 non-null  object
 5   overview              44078 non-null  object
 6   popularity            44998 non-null  object
 7   production_companies  44998 non-null  object
 8   production_countries  44998 non-null  object
 9   release_date          44920 non-null  object
 10  runtime               44747 non-null  float64
 11  title                 44998 non-null  object
 12  video                 44998 non-null  object
 13  vote_average          44998 non-null  float64
 14  vote_count            44998 non-null  float64
dtypes: float64(3), object(12)
memory usage: 5.5+ MB
```

In [43]:
```
# adult should be a boolean variable
movie['adult'].unique()
```

Out[43]:
```
array(['False', 'True'], dtype=object)
```

In [44]:
```
# remove data
movie.drop(movie[movie.adult==' - Written by Ørnås'].index,inplace=True)
movie.drop(movie[movie.adult==' Rune Balot goes to a casino connected to the O
ctober corporation to try to wrap up her case once and for all.'].index,inplac
e=True)
movie.drop(movie[movie.adult==' Avalanche Sharks tells the story of a bikini c
ontest that turns into a horrifying affair when it is hit by a shark avalanch
e.'].index,inplace=True)
```

In [45]:
```
movie.astype({'adult': 'bool','id':'int64','popularity':'float64'}).dtypes
```

Out[45]:
```
adult                      bool
genres                   object
id                        int64
imdb_id                  object
original_language        object
overview                 object
popularity              float64
production_companies     object
production_countries     object
release_date             object
runtime                 float64
title                    object
video                    object
vote_average            float64
vote_count              float64
dtype: object
```

In [46]:
```
movie.drop(movie[movie.popularity=='Beware Of Frost Bites'].index,inplace=True
)
```

In [47]:
```
# change dtype of release date
movie.astype({'release_date':'datetime64[ns]'}).dtypes
```

Out[47]:
```
adult                           object
genres                          object
id                              object
imdb_id                         object
original_language               object
overview                        object
popularity                      object
production_companies            object
production_countries            object
release_date            datetime64[ns]
runtime                        float64
title                           object
video                           object
vote_average                   float64
vote_count                     float64
dtype: object
```

In [48]:
```
movie['original_language'].unique()
```

Out[48]:
```
array(['en', 'fr', 'zh', 'it', 'fa', 'nl', 'de', 'cn', 'ar', 'es', 'ru',
       'sv', 'ja', 'ko', 'sr', 'bn', 'he', 'pt', 'wo', 'ro', 'hu', 'cy',
       'vi', 'cs', 'da', 'no', 'nb', 'pl', 'el', 'sh', 'xx', 'mk', 'bo',
       'ca', 'fi', 'th', 'sk', 'bs', 'hi', 'tr', 'is', 'ps', 'ab', 'eo',
       'ka', 'mn', 'bm', 'zu', 'uk', 'af', 'la', 'et', 'ku', 'fy', 'lv',
       'ta', 'sl', 'tl', 'ur', 'rw', 'id', 'bg', 'mr', 'lt', 'kk', 'ms',
       'sq', nan, 'qu', 'te', 'am', 'jv', 'tg', 'ml', 'hr', 'lo', 'ay',
       'kn', 'eu', 'ne', 'pa', 'ky', 'gl', 'uz', 'sm', 'mt', 'hy', 'iu',
       'lb', 'si'], dtype=object)
```

In [49]:
```
# change dtype of release date
movie.astype({'original_language':'category'}).dtypes
```

Out[49]:
```
adult                    object
genres                   object
id                       object
imdb_id                  object
original_language      category
overview                 object
popularity               object
production_companies     object
production_countries     object
release_date             object
runtime                 float64
title                    object
video                    object
vote_average            float64
vote_count              float64
dtype: object
```

In [50]:
```
movie.sample(3)
```

Out[50]:

	adult	genres	id	imdb_id	original_language	overview	popularity	production_c
1449	False	[{'id': 878, 'name': 'Science Fiction'}, {'id'...	10357	tt0120461	en	An earthquake shatters a peaceful Los Angeles ...	13.147917	[{'name' Centu
969	False	[{'id': 18, 'name': 'Drama'}, {'id': 28, 'name...	9400	tt0117603	en	Four black women, all of whom have suffered fo...	6.85606	[{'name' Cinem
39830	False	[{'id': 80, 'name': 'Crime'}, {'id': 9648, 'na...	231811	tt2183070	sv	On a crystal clear, starry and bitingly cold w...	1.986595	[{'name': ' Produktion A

handle missing data

In [51]:
```
movie.isnull().sum()
```

Out[51]:
```
adult                    0
genres                   0
id                       0
imdb_id                 15
original_language       10
overview               920
popularity               0
production_companies     0
production_countries     0
release_date            78
runtime                251
title                    0
video                    0
vote_average             0
vote_count               0
dtype: int64
```

In [52]:
```
movie.drop(columns=['overview'],inplace=True)
```

In [53]:
```
def clean(data):
    nan_value = np.nan
    # find out missing values in quote
    data.replace("", nan_value, inplace=True)
    data.replace(" ", nan_value, inplace=True)

    # find out missing values in fresh
    data.replace("none", nan_value, inplace=True)

    # clean fresh and quote by dropping N/As
    data.dropna(subset=['id','imdb_id','title'],inplace=True)

    return data
```

In [54]:
```
clean(movie)
```

Out[54]:

	adult	genres	id	imdb_id	original_language	popularity	production_companie
0	False	[{'id': 16, 'name': 'Animation'}, {'id': 35, '...	862	tt0114709	en	21.946943	[{'name': 'Pixa Animation Studios', 'id 3
1	False	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...	8844	tt0113497	en	17.015539	[{'name': 'TriSta Pictures', 'id': 559 {'na.
2	False	[{'id': 10749, 'name': 'Romance'}, {'id': 35, ...	15602	tt0113228	en	11.7129	[{'name': 'Warner Bros. 'id': 6194}, {'name'.
3	False	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'nam...	31357	tt0114885	en	3.859495	[{'name': 'Twentiet Century Fox Filr Corporat.
4	False	[{'id': 35, 'name': 'Comedy'}]	11862	tt0113041	en	8.387519	[{'name': 'Sandolla Productions', 'id 5842}.
...	...	...	...	...	...	...	.
45461	False	[{'id': 18, 'name': 'Drama'}, {'id': 10751, 'n...	439050	tt6209470	fa	0.072051	
45462	False	[{'id': 18, 'name': 'Drama'}]	111109	tt2028550	tl	0.178241	[{'name': 'Sine Olivia 'id': 19653
45463	False	[{'id': 28, 'name': 'Action'}, {'id': 18, 'nam...	67758	tt0303758	en	0.903007	[{'name': 'America World Pictures', 'id 6165
45464	False	[]	227506	tt0008536	en	0.003503	[{'name': 'Yermoliev', 'id 88753
45465	False	[]	461257	tt6980792	en	0.163015	

44983 rows × 14 columns

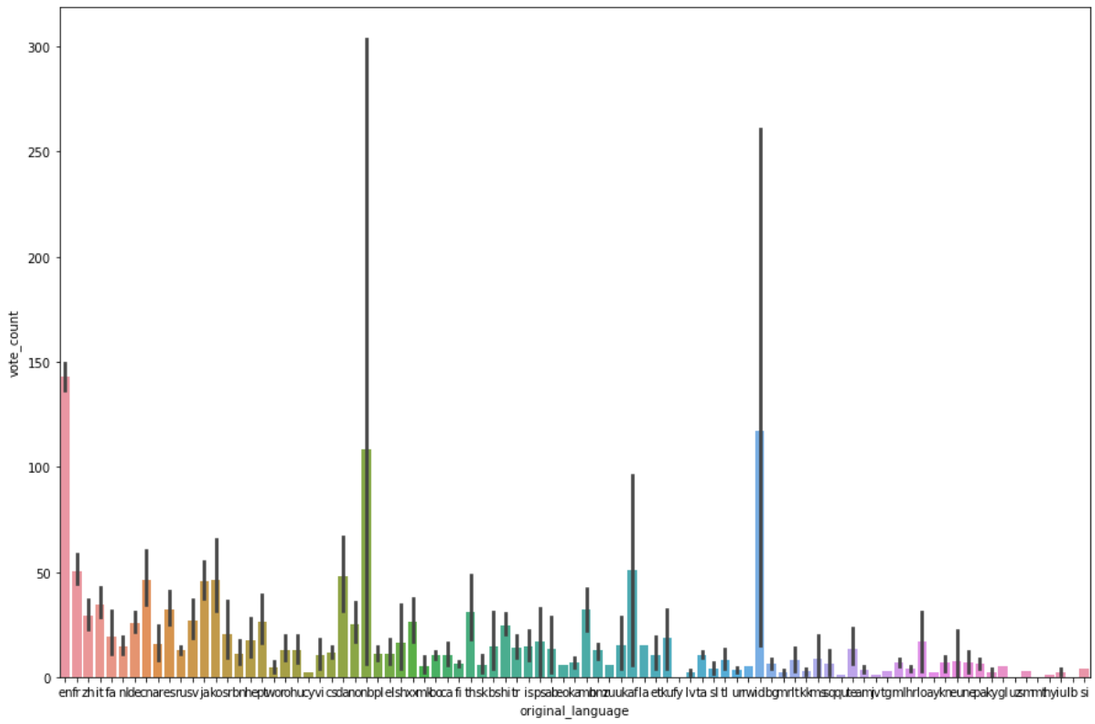
split datasets

In [55]:
```
# genres
genre = movie[["id","genres"]]
# production companies
production_company = movie[['id','production_companies']]
# production countries
production_country = movie[['id','production_countries']]
# remove these columns in movie dataset
movie=movie.drop(columns=['genres','production_companies','production_countries'])
```

typos or inconsistent capitalization

In [56]:
```
a4_dims = (15, 10)
fig, ax = pyplot.subplots(figsize=a4_dims)
sns.barplot(ax=ax,x="original_language", y="vote_count", data=movie)
```

Out[56]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x20c8c5a8310>
```



export data

In [57]:
```
movie.to_csv('..\data\movie.csv',index=False)
genre.to_csv('..\data\genre.csv',index=False)
production_company.to_csv('..\data\production_company.csv',index=False)
production_country.to_csv('..\data\production_country.csv',index=False)
```