untitled1

June 26, 2024

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

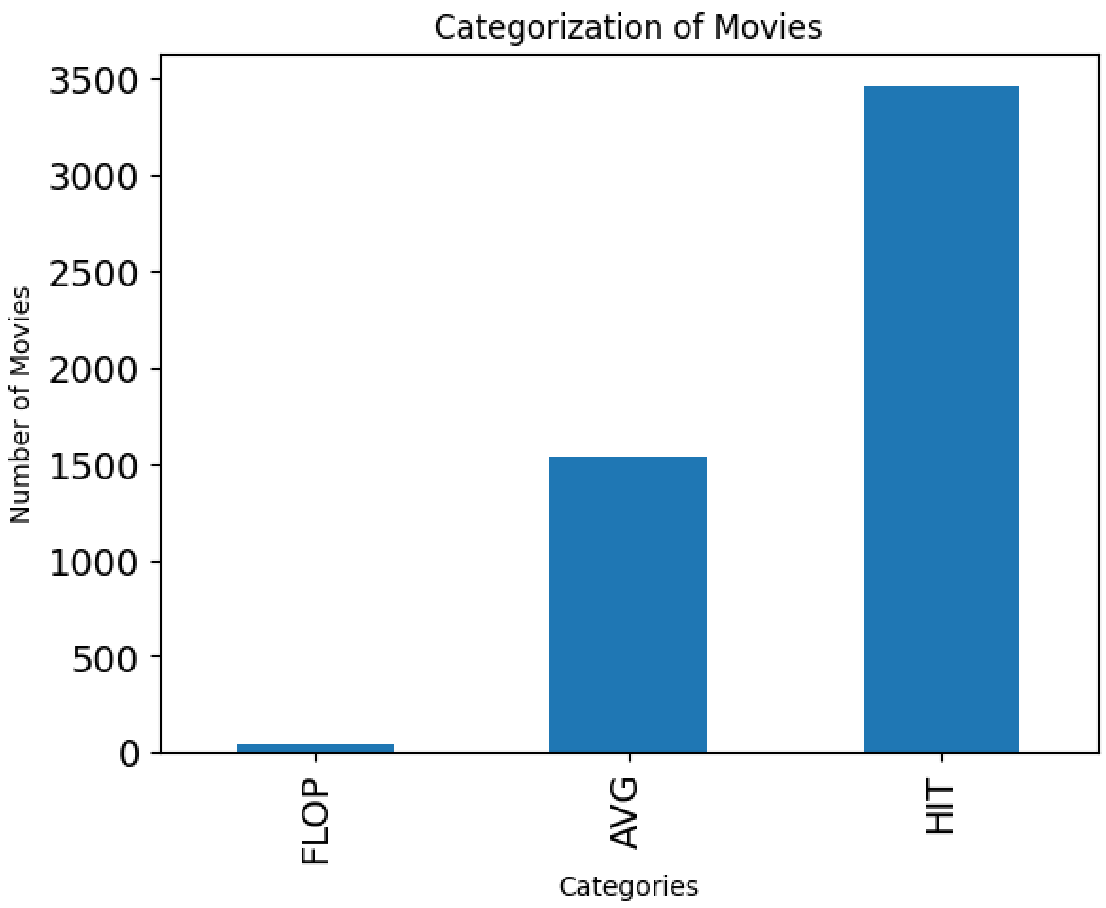
[5]:
```
data=pd.read_csv('movie_metadata.csv')
data.head()
```

[5]:
```
   color      director_name  num_critic_for_reviews  duration  \
0  Color      James Cameron                   723.0     178.0
1  Color     Gore Verbinski                   302.0     169.0
2  Color         Sam Mendes                   602.0     148.0
3  Color  Christopher Nolan                   813.0     164.0
4    NaN        Doug Walker                     NaN       NaN

   director_facebook_likes  actor_3_facebook_likes      actor_2_name  \
0                      0.0                   855.0  Joel David Moore
1                    563.0                  1000.0     Orlando Bloom
2                      0.0                   161.0      Rory Kinnear
3                  22000.0                 23000.0    Christian Bale
4                    131.0                     NaN        Rob Walker

   actor_1_facebook_likes        gross                           genres  …  \
0                  1000.0  760505847.0  Action|Adventure|Fantasy|Sci-Fi  …
1                 40000.0  309404152.0         Action|Adventure|Fantasy  …
2                 11000.0  200074175.0        Action|Adventure|Thriller  …
3                 27000.0  448130642.0                  Action|Thriller  …
4                   131.0          NaN                      Documentary  …

  num_user_for_reviews language  country  content_rating       budget  \
0               3054.0  English      USA           PG-13  237000000.0
1               1238.0  English      USA           PG-13  300000000.0
2                994.0  English       UK           PG-13  245000000.0
3               2701.0  English      USA           PG-13  250000000.0
4                  NaN      NaN      NaN             NaN          NaN

   title_year actor_2_facebook_likes imdb_score  aspect_ratio  \
0      2009.0                    936.0        7.9            1.78
1      2007.0                   5000.0        7.1            2.35
2      2015.0                    393.0        6.8            2.35
3      2012.0                  23000.0        8.5            2.35
4         NaN                     12.0        7.1             NaN

  movie_facebook_likes
0                33000
1                    0
2                85000
3               164000
4                    0

[5 rows x 28 columns]
```

[6]:
```
bins = [ 1, 3, 6, 10]
labels = ['FLOP', 'AVG', 'HIT']
data['imdb_binned'] = pd.cut(data['imdb_score'], bins=bins, labels=labels)
```

[7]:
```
data.groupby(['imdb_binned']).size().plot(kind="bar",fontsize=14)
plt.xlabel('Categories')
plt.ylabel('Number of Movies')
plt.title('Categorization of Movies')
```

[7]:
```
Text(0.5, 1.0, 'Categorization of Movies')
```



[8]:
```
data.head(5)
```

[8]:
```
   color      director_name  num_critic_for_reviews  duration  \
0  Color      James Cameron                   723.0     178.0
1  Color     Gore Verbinski                   302.0     169.0
2  Color         Sam Mendes                   602.0     148.0
3  Color  Christopher Nolan                   813.0     164.0
4    NaN        Doug Walker                     NaN       NaN

   director_facebook_likes  actor_3_facebook_likes      actor_2_name  \
0                      0.0                   855.0  Joel David Moore
1                    563.0                  1000.0     Orlando Bloom
2                      0.0                   161.0      Rory Kinnear
3                  22000.0                 23000.0    Christian Bale
4                    131.0                     NaN        Rob Walker

   actor_1_facebook_likes        gross                           genres  …  \
0                  1000.0  760505847.0  Action|Adventure|Fantasy|Sci-Fi  …
```

```
1                 40000.0  309404152.0        Action|Adventure|Fantasy  …
2                 11000.0  200074175.0       Action|Adventure|Thriller  …
3                 27000.0  448130642.0                 Action|Thriller  …
4                   131.0          NaN                     Documentary  …

  language country  content_rating       budget title_year  \
0  English     USA           PG-13  237000000.0     2009.0
1  English     USA           PG-13  300000000.0     2007.0
2  English      UK           PG-13  245000000.0     2015.0
3  English     USA           PG-13  250000000.0     2012.0
4      NaN     NaN             NaN          NaN        NaN

   actor_2_facebook_likes imdb_score aspect_ratio  movie_facebook_likes  \
0                   936.0        7.9         1.78                 33000
1                  5000.0        7.1         2.35                     0
2                   393.0        6.8         2.35                 85000
3                 23000.0        8.5         2.35                164000
4                    12.0        7.1          NaN                     0

  imdb_binned
0         HIT
1         HIT
2         HIT
3         HIT
4         HIT

[5 rows x 29 columns]
```

[9]:
```
data.shape
```

[9]:
```
(5043, 29)
```

[10]:
```
data.isnull().sum()
```

[10]:
```
color                         19
director_name                104
num_critic_for_reviews        50
duration                      15
director_facebook_likes      104
actor_3_facebook_likes        23
actor_2_name                  13
actor_1_facebook_likes         7
gross                        884
genres                         0
actor_1_name                   7
movie_title                    0
num_voted_users                0
```

```
cast_total_facebook_likes      0
actor_3_name                  23
facenumber_in_poster          13
plot_keywords                153
movie_imdb_link                0
num_user_for_reviews          21
language                      14
country                        5
content_rating               303
budget                       492
title_year                   108
actor_2_facebook_likes        13
imdb_score                     0
aspect_ratio                 329
movie_facebook_likes           0
imdb_binned                    0
dtype: int64
```

[11]:
```
data.dropna(inplace=True)
```

[12]:
```
data.shape
```

[12]:
```
(3755, 29)
```

[13]:
```
data.columns
```

[13]:
```
Index(['color', 'director_name', 'num_critic_for_reviews', 'duration',
       'director_facebook_likes', 'actor_3_facebook_likes', 'actor_2_name',
       'actor_1_facebook_likes', 'gross', 'genres', 'actor_1_name',
       'movie_title', 'num_voted_users', 'cast_total_facebook_likes',
       'actor_3_name', 'facenumber_in_poster', 'plot_keywords',
       'movie_imdb_link', 'num_user_for_reviews', 'language', 'country',
       'content_rating', 'budget', 'title_year', 'actor_2_facebook_likes',
       'imdb_score', 'aspect_ratio', 'movie_facebook_likes', 'imdb_binned'],
      dtype='object')
```

[14]:
```
data.shape
```

[14]:
```
(3755, 29)
```

[15]:
```
data.describe(include='object')
```

[15]:
```
        color      director_name     actor_2_name              genres  \
count    3755               3755             3755                3755
unique      2               1658             2187                 745
top     Color  Steven Spielberg  Morgan Freeman  Comedy|Drama|Romance
freq     3631                 25               20                 147
```

```
          actor_1_name           movie_title  actor_3_name  \
count             3755                  3755          3755
unique            1427                  3654          2586
top     Robert De Niro  Victor Frankenstein   Steve Coogan
freq                42                     3             8

                                           plot_keywords  \
count                                               3755
unique                                              3655
top     halloween|masked killer|michael myers|slasher|…
freq                                                   3

                                           movie_imdb_link language country  \
count                                                 3755     3755    3755
unique                                                3655       33      45
top     http://www.imdb.com/title/tt0077651/?ref_=fn_t…  English     USA
freq                                                   3     3598    2986

       content_rating
count            3755
unique             12
top                 R
freq             1700
```

[16]:
```
data.drop(columns=['movie_title','movie_imdb_link'],inplace=True)
```

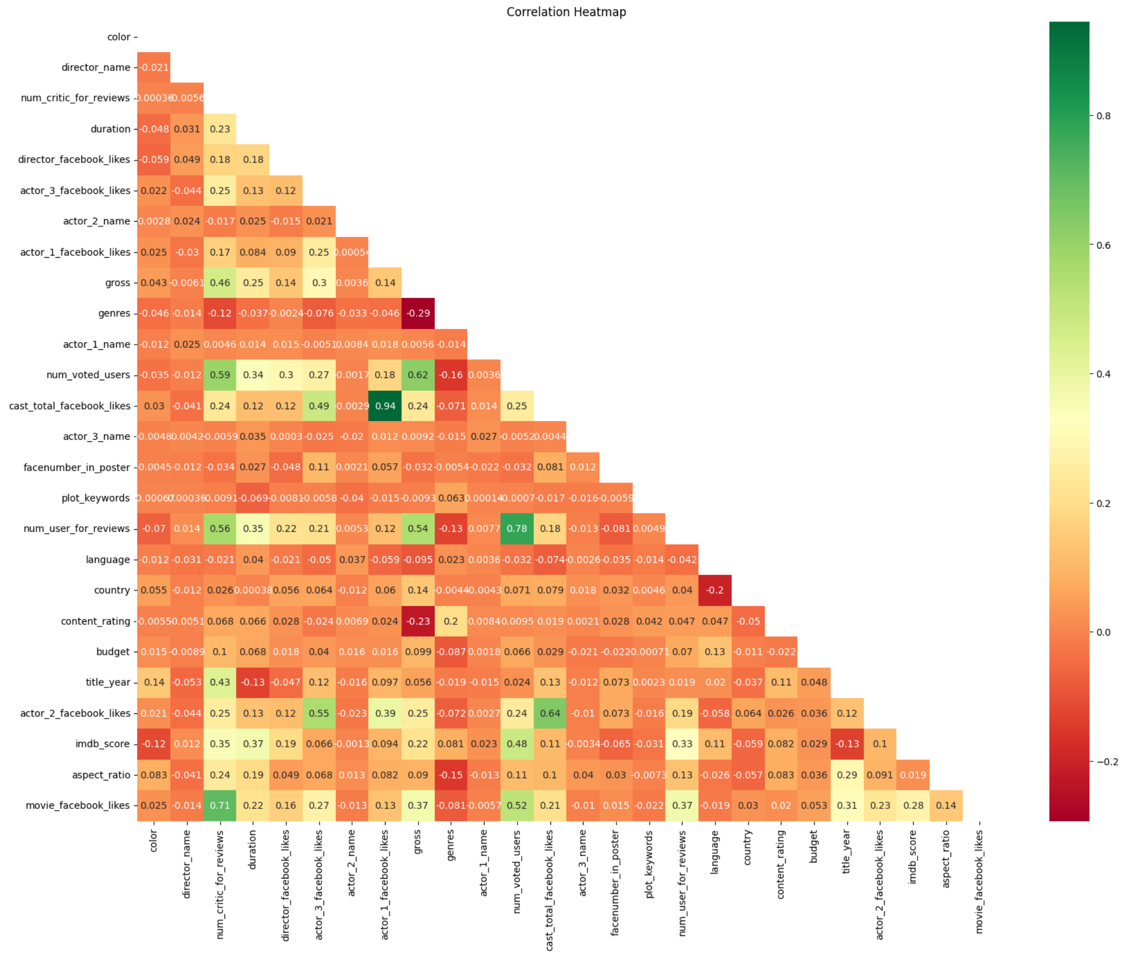
[18]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
cat_list=['color', 'director_name', 'actor_2_name',
        'genres', 'actor_1_name',
        'actor_3_name',
        'plot_keywords',
        'language', 'country', 'content_rating',
       'title_year', 'aspect_ratio']
data[cat_list]=data[cat_list].apply(lambda x:le.fit_transform(x))
```

[19]:
```
data.head()
```

[19]:
```
   color  director_name  num_critic_for_reviews  duration  \
0      1            620                   723.0     178.0
1      1            538                   302.0     169.0
2      1           1394                   602.0     148.0
3      1            251                   813.0     164.0
5      1             62                   462.0     132.0

   director_facebook_likes  actor_3_facebook_likes  actor_2_name  \
0                     0.0                    855.0          1001
1                   563.0                   1000.0          1591
2                     0.0                    161.0          1794
3                 22000.0                  23000.0           380
5                   475.0                    530.0          1836

   actor_1_facebook_likes        gross  genres  …  language  country  \
0                  1000.0  760505847.0      91  …         9       43
1                 40000.0  309404152.0      85  …         9       43
2                 11000.0  200074175.0     107  …         9       42
3                 27000.0  448130642.0     243  …         9       43
5                   640.0   73058679.0     105  …         9       43

   content_rating       budget  title_year  actor_2_facebook_likes  \
0               7  237000000.0          66                   936.0
1               7  300000000.0          64                  5000.0
2               7  245000000.0          72                   393.0
3               7  250000000.0          69                 23000.0
5               7  263700000.0          69                   632.0

   imdb_score  aspect_ratio  movie_facebook_likes  imdb_binned
0         7.9             7                 33000          HIT
1         7.1            12                     0          HIT
2         6.8            12                 85000          HIT
3         8.5            12                164000          HIT
5         6.6            12                 24000          HIT

[5 rows x 27 columns]
```

[24]:
```
numeric_data = data.select_dtypes(include=np.number)

# Compute the correlation matrix
corr = numeric_data.corr()

# Create a mask to hide the upper triangle of the plot
mask = np.zeros_like(corr, dtype=bool)
mask[np.triu_indices_from(mask)] = True

# Setting up the matplotlib figure
plt.figure(figsize=(20, 15))

# Plotting the heatmap using seaborn
sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns,␣
 ↪cmap='RdYlGn', annot=True, mask=mask)

# Adding title to the plot
plt.title('Correlation Heatmap')
```

```
# Displaying the plot
plt.show()
```



[25]:
```
data.
 ↪drop(columns=['cast_total_facebook_likes','num_critic_for_reviews'],inplace=True)
```

[26]:
```
data.drop(columns=['imdb_score'],inplace=True)
```

[27]:
```
data.shape
```

[27]:
```
(3755, 24)
```

[30]:
```
X = data.iloc[:, 0:23].values
#Dependent/Target Variable
y = data.iloc[:, 23].values
y
```

[30]:
```
['HIT', 'HIT', 'HIT', 'HIT', 'HIT', …, 'HIT', 'HIT', 'HIT', 'HIT', 'HIT']
Length: 3755
Categories (3, object): ['FLOP' < 'AVG' < 'HIT']
```

[32]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,␣
 ↪random_state = 0,stratify = y)
print(X_train.shape)
print(y_train.shape)
```

```
(2628, 23)
(2628,)
```

[33]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

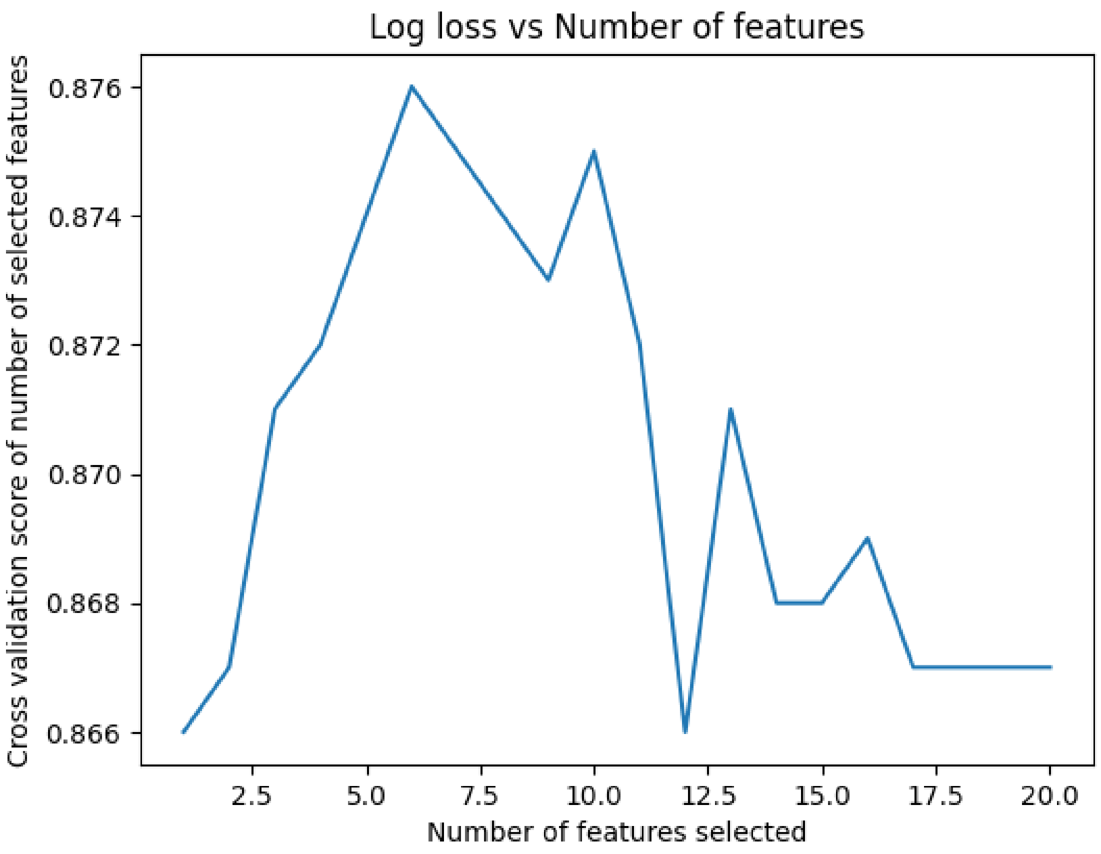
[34]:
```
from sklearn.feature_selection import RFECV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import log_loss
clf_rf=RandomForestClassifier(random_state=0)
rfecv=RFECV(estimator=clf_rf, step=1,cv=5,scoring='neg_log_loss')
rfecv=rfecv.fit(X_train,y_train)
```

[35]:
```
X_train = pd.DataFrame(X_train)
X_test = pd.DataFrame(X_test)
print('Optimal number of features :', rfecv.n_features_)
print('Best features :', X_train.columns[rfecv.support_])
```

```
Optimal number of features : 22
Best features : Index([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
17, 18, 19, 20,
       21, 22],
      dtype='int64')
```

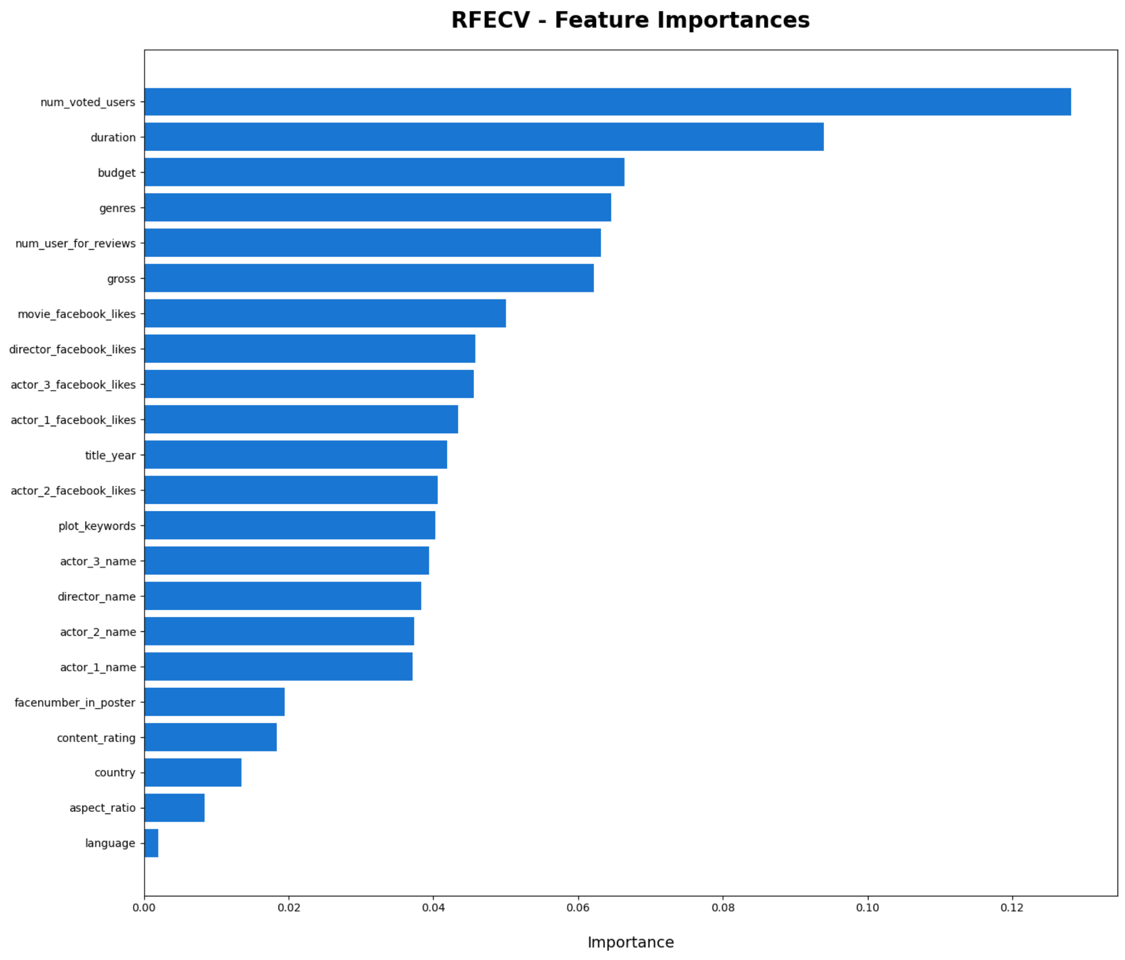
[36]:
```
clf_rf = clf_rf.fit(X_train,y_train)
importances = clf_rf.feature_importances_

std = np.std([tree.feature_importances_ for tree in clf_rf.estimators_],
             axis=0)
indices = np.argsort(importances)[::-1]
```

[63]:
```
import matplotlib.pyplot as plt

plt.figure()
plt.xlabel("Number of features selected")
plt.ylabel("Cross validation score of number of selected features")
plt.title("Log loss vs Number of features")
plt.plot(range(1, len(rfecv.cv_results_['mean_test_score']) + 1), rfecv.
 ↪cv_results_['mean_test_score'])
plt.show()
```



[70]:
```
# Ensure that we are working with the data1 containing the correct features
data1 = data.copy()

# Drop the 'imdb_binned' column if it exists
if 'imdb_binned' in data1.columns:
    data1.drop(columns=['imdb_binned'], inplace=True)

# Selecting the Important Features
X_opt_train = rfecv.transform(X_train)
X_opt_test = rfecv.transform(X_test)

# Scaling the selected features
sc = StandardScaler()
X_opt_train = sc.fit_transform(X_opt_train)
```

```python
X_opt_test = sc.transform(X_opt_test)

# Creating a new dataframe with column names and feature importance
dset = pd.DataFrame()
dset['attr'] = data1.columns[rfecv.support_]
dset['importance'] = rfecv.estimator_.feature_importances_

# Sorting with importance column
dset = dset.sort_values(by='importance', ascending=True)

# Barplot indicating Feature Importance
plt.figure(figsize=(16, 14))
plt.barh(y=dset['attr'], width=dset['importance'], color='#1976D2')
plt.title('RFECV - Feature Importances', fontsize=20, fontweight='bold', pad=20)
plt.xlabel('Importance', fontsize=14, labelpad=20)
plt.show()
```



[90]:
```
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy',␣
 ↪random_state = 0)
classifier.fit(X_opt_train,y_train)
```

[90]:
```
RandomForestClassifier(criterion='entropy', random_state=0)
```

[92]:
```
y_pred = classifier.predict(X_opt_test)
```

[93]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

[93]:
```
array([[191,   0, 144],
       [  4,   0,   5],
       [ 68,   0, 715]])
```

[94]:
```
from sklearn.metrics import classification_report
cr = classification_report(y_test,y_pred)
print(cr)
```

```
              precision    recall  f1-score   support

         AVG       0.73      0.57      0.64       335
        FLOP       0.00      0.00      0.00         9
         HIT       0.83      0.91      0.87       783

    accuracy                           0.80      1127
   macro avg       0.52      0.49      0.50      1127
weighted avg       0.79      0.80      0.79      1127

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```