In [1]:
```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split

# Load the dataset
data = pd.read_csv('Downloads/anime_rating.csv')
```

In [2]:
```
data.head()
```

Out[2]:

	title	mediaType	eps	duration	startYr	finishYr	description	contentWarn	wat
0	Dragon Ball Z Movie 15: Resurrection 'F'	Movie	1	67.0	2015	2015	Even the complete obliteration of his physical...	No	
1	Kuripuri*Kuripura	Movie	1	5.0	2008	2008	NaN	No	
2	GJ-bu@	TV Special	1	46.0	2014	2014	The story is set during the spring vacation im...	No	
3	Nausicaa of the Valley of the Wind	Movie	1	67.0	1984	1984	One thousand years after the Giant Warriors ca...	No	
4	Miru Tights Special	DVD Special	1	4.0	2019	2019	Yua is asked to model an illustrator's designs...	No	

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6523 entries, 0 to 6522
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   title           6523 non-null   object 
 1   mediaType       6496 non-null   object 
 2   eps             6523 non-null   int64  
 3   duration        6248 non-null   float64
 4   startYr         6523 non-null   int64  
 5   finishYr        6523 non-null   int64  
 6   description     4114 non-null   object 
 7   contentWarn     6523 non-null   object 
 8   watched         6523 non-null   int64  
 9   watching        6523 non-null   int64  
 10  rating          6523 non-null   float64
 11  votes           6496 non-null   float64
 12  studio_primary  6523 non-null   object 
 13  studios_colab   6523 non-null   object 
 14  genre           6523 non-null   object 
dtypes: float64(3), int64(5), object(7)
memory usage: 764.5+ KB
```

In [4]:
```
data.describe()
```

Out[4]:

	eps	duration	startYr	finishYr	watched	watching	r
count	6523.000000	6248.000000	6523.000000	6523.000000	6523.000000	6523.000000	6523.00
mean	8.716235	18.396287	2005.241147	2005.575349	1347.948643	57.445654	2.96
std	11.002479	20.949350	12.911035	12.568169	1737.138112	76.527405	0.76
min	1.000000	1.000000	1967.000000	1970.000000	5.000000	0.000000	1.11
25%	1.000000	5.000000	2000.000000	2000.000000	56.000000	2.000000	2.37
50%	1.000000	7.000000	2010.000000	2010.000000	349.000000	13.000000	2.94
75%	12.000000	25.000000	2015.000000	2015.000000	2252.500000	98.000000	3.56
max	34.000000	67.000000	2020.000000	2020.000000	4649.000000	199.000000	4.70

In [5]:
```
# Step 1: Data Cleaning
# Handle missing values
data['duration'] = data['duration'].fillna(data['duration'].median())
data['mediaType'] = data['mediaType'].fillna('Unknown')  # Fill missi
data['votes'] = data['votes'].fillna(data['votes'].median())  # Fill m
data.drop(columns=['description'], inplace=True)  # Drop 'description'
```

In [6]:
```
# Step 2: Encoding categorical variables
le = LabelEncoder()
categorical_features = ['mediaType', 'contentWarn', 'studio_primary',
for feature in categorical_features:
    data[feature] = le.fit_transform(data[feature])
```

In [7]:
```
# Step 3: Feature Scaling
scaler = StandardScaler()
numerical_features = ['eps', 'duration', 'watched', 'watching', 'votes
data[numerical_features] = scaler.fit_transform(data[numerical_feature
```

In [8]:
```
# Step 4: Prepare data for OLS modeling
X = data.drop(columns=['title', 'rating'])  # Exclude 'title' and targ
y = data['rating']
```

In [9]:
```

# Add a constant term to the predictors for OLS
X = sm.add_constant(X)

# Step 5: Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.
```

In [10]:

```
# Step 6: Fit OLS model
ols_model = sm.OLS(y_train, X_train).fit()

# Step 7: Evaluate the OLS model
print(ols_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 rating   R-squared:                       0.696
Model:                            OLS   Adj. R-squared:                  0.695
Method:                 Least Squares   F-statistic:                     993.5
Date:                Tue, 19 Nov 2024   Prob (F-statistic):               0.00
Time:                        16:49:11   Log-Likelihood:                -2847.2
No. Observations:                5218   AIC:                             5720.
Df Residuals:                    5205   BIC:                             5806.
Df Model:                          12
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        -13.1223      1.107    -11.854      0.000     -15.293     -10.952
mediaType     -0.0342      0.003    -11.293      0.000      -0.040      -0.028
eps            0.1778      0.008     22.740      0.000       0.162       0.193
duration       0.2775      0.007     39.885      0.000       0.264       0.291
startYr        0.0270      0.008      3.585      0.000       0.012       0.042
finishYr      -0.0189      0.008     -2.450      0.014      -0.034      -0.004
contentWarn   -0.1544      0.020     -7.866      0.000      -0.193      -0.116
watched        0.8168      0.047     17.533      0.000       0.725       0.908
watching       0.2208      0.013     17.360      0.000       0.196       0.246
votes         -0.4910      0.051     -9.695      0.000      -0.590      -0.392
studio_primary     0.0197      0.003      6.598      0.000       0.014       0.026
studios_colab    -0.0096      0.028     -0.348      0.728      -0.064       0.045
genre         -0.0017      0.004     -0.380      0.704      -0.010       0.007
==============================================================================
Omnibus:                      121.485   Durbin-Watson:                   2.033
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               63.091
Skew:                           0.022   Prob(JB):                     2.00e-14
Kurtosis:                       2.463   Cond. No.                     5.42e+05
```

```
==========
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
[2] The condition number is large, 5.42e+05. This might indicate tha
t there are
strong multicollinearity or other numerical problems.
```

In [11]:
```
# Step 8: Predictions on the test set
y_pred = ols_model.predict(X_test)
```

In [12]:
```
# Calculate evaluation metrics
from sklearn.metrics import mean_squared_error, mean_absolute_error, r

rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"OLS Regression - RMSE: {rmse}, MAE: {mae}, R²: {r2}")
```

```
OLS Regression - RMSE: 0.42742004031158903, MAE: 0.3506061630699016
7, R²: 0.6932291747504539
```