```
In [ ]: import pandas as pd
        from sklearn.model_selection import train_test_split
        from sklearn.feature_extraction.text import TfidfVectorizer
        from sklearn.linear_model import LinearRegression
        from sklearn.metrics import mean_squared_error, r2_score
        import re

        # Load the dataset
        file_name = 'TheOfficeImdb.csv'
        office_df = pd.read_csv(file_name)

        # Select relevant columns
        office_df = office_df[['season', 'episode_num', 'desc', 'imdb_rating']]

        # Text preprocessing function
        def preprocess_text(text):
            text = text.lower()
            text = re.sub(r'\d+', '', text)
            text = re.sub(r'[^\w\s]', '', text)
            text = re.sub(r'\s+', ' ', text).strip()
            return text

        # Apply text preprocessing
        office_df['desc'] = office_df['desc'].apply(preprocess_text)

        # Split the data into features and target variable
        X = office_df[['season', 'episode_num', 'desc']]
        y = office_df['imdb_rating']

        # Vectorize the text data
        vectorizer = TfidfVectorizer(max_features=5000)
        X_desc_vect = vectorizer.fit_transform(X['desc'])

        # Combine text vectors with other features
        X_other = X[['season', 'episode_num']].reset_index(drop=True)
        X_combined = pd.concat([pd.DataFrame(X_desc_vect.toarray()), X_other], axis=1)

        # Convert all column names to strings
        X_combined.columns = X_combined.columns.astype(str)

        # Split the data into training and testing sets
        X_train, X_test, y_train, y_test = train_test_split(X_combined, y, test_size=0.2, r

        # Create a Linear Regression model
        model = LinearRegression()

        # Train the model
        model.fit(X_train, y_train)

        # Make predictions
        y_pred = model.predict(X_test)

        # Evaluate the model
        mse = mean_squared_error(y_test, y_pred)
```

```
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

```
Mean Squared Error: 0.2936179433508566
R-squared: 0.1213852988257692
```

In [ ]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 188 entries, 0 to 187
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   season             188 non-null    int64  
 1   episode_num        188 non-null    int64  
 2   title              188 non-null    object 
 3   original_air_date  188 non-null    object 
 4   imdb_rating        188 non-null    float64
 5   total_votes        188 non-null    int64  
 6   desc               188 non-null    object 
dtypes: float64(1), int64(3), object(3)
memory usage: 10.4+ KB
```

In [ ]: