In [1]:
```
import pandas as pd
import os
from pathlib import Path
import zipfile
```

In [2]:
```
# --- Step 1: Extract Netflix and Baby Name ZIP Files ---

netflix_path = "C:\\Users\\himav\\OneDrive\\Desktop\\uni\\big data project\\netf
female_zip_path = "C:\\Users\\himav\\OneDrive\\Desktop\\uni\\big data project\\b
male_zip_path = "C:\\Users\\himav\\OneDrive\\Desktop\\uni\\big data project\\bab

female_extract_dir = "C:\\Users\\himav\\OneDrive\\Desktop\\uni\\big data project
male_extract_dir = "C:\\Users\\himav\\OneDrive\\Desktop\\uni\\big data project\\

with zipfile.ZipFile(female_zip_path, 'r') as zip_ref:
    zip_ref.extractall(female_extract_dir)

with zipfile.ZipFile(male_zip_path, 'r') as zip_ref:
    zip_ref.extractall(male_extract_dir)

female_nested_dir = os.path.join(female_extract_dir, "female")
male_nested_dir = os.path.join(male_extract_dir, "male")

female_csv_files = sorted([f for f in os.listdir(female_nested_dir) if f.endswit
male_csv_files = sorted([f for f in os.listdir(male_nested_dir) if f.endswith(".
```

In [3]:
```
# --- Step 2: Load and Clean Baby Name Data ---

def load_baby_name_file(filepath, gender):
    try:
        df = pd.read_csv(filepath, encoding='utf-8')
    except UnicodeDecodeError:
        df = pd.read_csv(filepath, encoding='ISO-8859-1')
    name_col = 'Given Name' if 'Given Name' in df.columns else 'First Name'
    year = int(Path(filepath).stem[-4:])
    df[name_col] = df[name_col].str.lower().str.strip()
    df['Year'] = year
    df['Gender'] = gender
    df = df.rename(columns={name_col: 'Name'})
    return df[['Name', 'Amount', 'Year', 'Gender']]

male_dfs = [load_baby_name_file(os.path.join(male_nested_dir, file), 'Male') for
female_dfs = [load_baby_name_file(os.path.join(female_nested_dir, file), 'Female

baby_names_all = pd.concat(male_dfs + female_dfs, ignore_index=True)

print("male dfs")
print(male_dfs)
print("female dfs")
print(female_dfs)
print("baby names all")
print(baby_names_all)
```

```
male dfs
[         Name  Amount  Year Gender
0      oliver     184  2014   Male
1        jack     132  2014   Male
2     william     128  2014   Male
3       lucas     124  2014   Male
4        noah     112  2014   Male
...       ...     ...   ...    ...
2437   zoylin       1  2014   Male
2438   zubair       1  2014   Male
2439   zuriel       1  2014   Male
2440    zylen       1  2014   Male
2441    zyrus       1  2014   Male

[2442 rows x 4 columns],            Name  Amount  Year Gender
0        oliver     186  2015   Male
1       william     134  2015   Male
2          jack     125  2015   Male
3          noah     123  2015   Male
4       charlie     103  2015   Male
...         ...     ...   ...    ...
2441       zoki       1  2015   Male
2442  zosailian       1  2015   Male
2443       zyah       1  2015   Male
2444       zyan       1  2015   Male
2445     zylenn       1  2015   Male

[2446 rows x 4 columns],          Name  Amount  Year Gender
0      oliver   190.0  2016   Male
1        jack   129.0  2016   Male
2     william   117.0  2016   Male
3       james   100.0  2016   Male
4       mason   100.0  2016   Male
...       ...     ...   ...    ...
2450  zorawar     1.0  2016   Male
2451   zuhair     1.0  2016   Male
2452   zuriel     1.0  2016   Male
2453      NaN     NaN  2016   Male
2454    total     NaN  2016   Male

[2455 rows x 4 columns],          Name  Amount  Year Gender
0      oliver     172  2017   Male
1     william     114  2017   Male
2        jack     111  2017   Male
3        noah     100  2017   Male
4       henry      89  2017   Male
...       ...     ...   ...    ...
2444    ziyan       1  2017   Male
2445   zovian       1  2017   Male
2446   zuhayr       1  2017   Male
2447   zuwaid       1  2017   Male
2448     zyon       1  2017   Male

[2449 rows x 4 columns],         Name  Amount  Year Gender
0     oliver     173  2018   Male
1       jack     132  2018   Male
2    william     117  2018   Male
3   harrison     115  2018   Male
4    charlie     105  2018   Male
..       ...     ...   ...    ...
```

```
95      beau      19  2018   Male
96   charles      19  2018   Male
97      ezra      19  2018   Male
98     jesse      19  2018   Male
99     wyatt      19  2018   Male

[100 rows x 4 columns],        Name  Amount  Year Gender
0    oliver     152  2019   Male
1       leo     106  2019   Male
2   william     102  2019   Male
3      jack     101  2019   Male
4      noah     101  2019   Male
..      ...     ...   ...    ...
95    billy      20  2019   Male
96   elliot      20  2019   Male
97     luka      20  2019   Male
98     luke      20  2019   Male
99  maxwell      20  2019   Male

[100 rows x 4 columns],        Name  Amount  Year Gender
0    oliver     147  2020   Male
1     henry     116  2020   Male
2      noah     107  2020   Male
3       leo     106  2020   Male
4   william     106  2020   Male
..      ...     ...   ...    ...
95     alby      18  2020   Male
96     axel      18  2020   Male
97    billy      18  2020   Male
98    chase      18  2020   Male
99    lenny      18  2020   Male

[100 rows x 4 columns],        Name  Amount  Year Gender
0    oliver     195  2021   Male
1      noah     132  2021   Male
2     henry     126  2021   Male
3   charlie     119  2021   Male
4       leo      98  2021   Male
..      ...     ...   ...    ...
95   joshua      21  2021   Male
96   julian      21  2021   Male
97   albert      20  2021   Male
98     axel      20  2021   Male
99   elliot      20  2021   Male

[100 rows x 4 columns],         Name  Amount  Year Gender
0     oliver     181  2022   Male
1       noah     142  2022   Male
2      henry     121  2022   Male
3        leo     108  2022   Male
4     archie     107  2022   Male
..       ...     ...   ...    ...
95  nicholas      21  2022   Male
96    oakley      21  2022   Male
97     ziggy      21  2022   Male
98       ari      20  2022   Male
99     billy      20  2022   Male

[100 rows x 4 columns],         Name  Amount  Year Gender
0     oliver     155  2023   Male
```

```
1      henry     112  2023   Male
2        leo     109  2023   Male
3       noah      92  2023   Male
4   theodore      87  2023   Male
..       ...     ...   ...    ...
95     ryder      19  2023   Male
96    alfred      18  2023   Male
97     elias      18  2023   Male
98   gabriel      18  2023   Male
99    huxley      18  2023   Male

[100 rows x 4 columns],           Name  Amount  Year Gender
0       oliver     131  2024   Male
1        henry     125  2024   Male
2         noah     105  2024   Male
3      charlie     101  2024   Male
4     theodore      98  2024   Male
...        ...     ...   ...    ...
2762      zyah       1  2024   Male
2763      zyan       1  2024   Male
2764     zylan       1  2024   Male
2765      zyne       1  2024   Male
2766     zyren       1  2024   Male

[2767 rows x 4 columns]]
female dfs
[           Name  Amount  Year  Gender
0     charlotte     128  2014  Female
1         grace     116  2014  Female
2         chloe     111  2014  Female
3        olivia     109  2014  Female
4         emily     106  2014  Female
...         ...     ...   ...     ...
2831    zuhanah       1  2014  Female
2832     zulika       1  2014  Female
2833      zunis       1  2014  Female
2834     zurine       1  2014  Female
2835    zyrinah       1  2014  Female

[2836 rows x 4 columns],            Name  Amount  Year  Gender
0     charlotte     124  2015  Female
1        amelia     111  2015  Female
2        olivia      97  2015  Female
3           ava      96  2015  Female
4      scarlett      93  2015  Female
...         ...     ...   ...     ...
2890      zosia       1  2015  Female
2891   zunairah       1  2015  Female
2892       zuri       1  2015  Female
2893    zuzanna       1  2015  Female
2894       zyla       1  2015  Female

[2895 rows x 4 columns],            Name  Amount  Year  Gender
0     charlotte   139.0  2016  Female
1        olivia   123.0  2016  Female
2           ava   116.0  2016  Female
3           mia   103.0  2016  Female
4        amelia    96.0  2016  Female
...         ...     ...   ...     ...
2799    zulayho     1.0  2016  Female
```

```
2800   zymeliah     1.0  2016  Female
2801        NaN     NaN  2016  Female
2802        NaN     NaN  2016  Female
2803      total     NaN  2016  Female

[2804 rows x 4 columns],            Name  Amount  Year  Gender
0     charlotte     132  2017  Female
1           ava     114  2017  Female
2          isla     110  2017  Female
3        harper      91  2017  Female
4        amelia      87  2017  Female
...         ...     ...   ...     ...
2759      ziyue       1  2017  Female
2760     zoraya       1  2017  Female
2761  zullfanah       1  2017  Female
2762     zuriel       1  2017  Female
2763    zuwayla       1  2017  Female

[2764 rows x 4 columns],          Name  Amount  Year  Gender
0   charlotte     119  2018  Female
1      amelia     114  2018  Female
2         ava     100  2018  Female
3         mia      97  2018  Female
4        isla      92  2018  Female
..        ...     ...   ...     ...
95      heidi      16  2018  Female
96    jasmine      16  2018  Female
97       luna      16  2018  Female
98     maggie      16  2018  Female
99      molly      16  2018  Female

[100 rows x 4 columns],          Name  Amount  Year  Gender
0   charlotte     119  2019  Female
1         ava     112  2019  Female
2      olivia     103  2019  Female
3       grace      93  2019  Female
4      amelia      86  2019  Female
..        ...     ...   ...     ...
95       lyla      17  2019  Female
96      paige      17  2019  Female
97       sage      17  2019  Female
98      sarah      17  2019  Female
99  alexandra      16  2019  Female

[100 rows x 4 columns],          Name  Amount  Year  Gender
0   charlotte     118  2020  Female
1      olivia     103  2020  Female
2      amelia      99  2020  Female
3         ava      96  2020  Female
4        isla      94  2020  Female
..        ...     ...   ...     ...
95       remi      18  2020  Female
96     amalia      17  2020  Female
97      clara      17  2020  Female
98       eden      17  2020  Female
99     hallie      17  2020  Female

[100 rows x 4 columns],          Name  Amount  Year  Gender
0        isla     120  2021  Female
1   charlotte     110  2021  Female
```

```
2      olivia     101  2021  Female
3      harper      95  2021  Female
4         ava      91  2021  Female
..        ...     ...   ...     ...
95    madison      20  2021  Female
96      maeve      20  2021  Female
97   savannah      19  2021  Female
98       lara      18  2021  Female
99       lola      18  2021  Female

[100 rows x 4 columns],          Name  Amount  Year  Gender
0        isla     111  2022  Female
1   charlotte     110  2022  Female
2      amelia      89  2022  Female
3      willow      84  2022  Female
4       grace      82  2022  Female
..        ...     ...   ...     ...
95      rylee      17  2022  Female
96    charlie      16  2022  Female
97       eden      16  2022  Female
98     hayley      16  2022  Female
99      piper      16  2022  Female

[100 rows x 4 columns],          Name  Amount  Year  Gender
0        isla     112  2023  Female
1   charlotte     104  2023  Female
2      olivia      86  2023  Female
3         ava      77  2023  Female
4         ivy      73  2023  Female
..        ...     ...   ...     ...
95   brooklyn      15  2023  Female
96     callie      15  2023  Female
97        gia      15  2023  Female
98    jasmine      15  2023  Female
99       nora      15  2023  Female

[100 rows x 4 columns],            Name  Amount  Year  Gender
0     charlotte     108  2024  Female
1        olivia      89  2024  Female
2          isla      86  2024  Female
3        amelia      70  2024  Female
4           mia      67  2024  Female
...         ...     ...   ...     ...
3025      zurwa       1  2024  Female
3026       zuzu       1  2024  Female
3027      zylah       1  2024  Female
3028      zylia       1  2024  Female
3029       zyra       1  2024  Female

[3030 rows x 4 columns]]
baby names all
          Name  Amount  Year  Gender
0       oliver   184.0  2014    Male
1         jack   132.0  2014    Male
2      william   128.0  2014    Male
3        lucas   124.0  2014    Male
4         noah   112.0  2014    Male
...        ...     ...   ...     ...
28083    zurwa     1.0  2024  Female
28084     zuzu     1.0  2024  Female
```

```
28085    zylah     1.0  2024  Female
28086    zylia     1.0  2024  Female
28087     zyra     1.0  2024  Female

[28088 rows x 4 columns]
```

In [4]:
```python
# --- Step 3: Clean Netflix Titles ---

netflix_df = pd.read_csv(netflix_path)
netflix_df = netflix_df.dropna(subset=['title', 'release_year'])
netflix_df['title'] = netflix_df['title'].str.lower().str.strip()
netflix_df = netflix_df[~netflix_df['title'].str.contains(" ")]
netflix_df = netflix_df.drop_duplicates(subset='title')
netflix_df = netflix_df[['title', 'release_year', 'type']]

print(netflix_df)
```

```
          title  release_year     type
2      ganglands          2021  TV Show
7        sankofa          1993    Movie
18     intrusion          2021    Movie
19        jaguar          2021  TV Show
24         jeans          1998    Movie
...          ...           ...      ...
8801     zinzana          2015    Movie
8802      zodiac          2007    Movie
8804  zombieland          2009    Movie
8805        zoom          2006    Movie
8806      zubaan          2015    Movie

[1630 rows x 3 columns]
```

In [5]:
```python
# --- Step 4: Label Influenced Titles ---

baby_names_by_year = baby_names_all.groupby('Year')['Name'].apply(set).to_dict()

def was_influenced(title, release_year):
    for year in range(release_year + 1, 2025):
        if year in baby_names_by_year and title in baby_names_by_year[year]:
            return 1
    return 0

netflix_df['influenced'] = netflix_df.apply(
    lambda row: was_influenced(row['title'], row['release_year']), axis=1
)
```

In [6]:
```python
# Final dataset is stored in netflix_df
print(netflix_df.head())
print()
print(netflix_df['influenced'].value_counts())
```

```
        title  release_year     type  influenced
2   ganglands          2021  TV Show           0
7     sankofa          1993    Movie           0
18  intrusion          2021    Movie           0
19     jaguar          2021  TV Show           0
24      jeans          1998    Movie           0

influenced
0    1500
1     130
Name: count, dtype: int64
```

In [7]:
```python
# Step 1: Feature Encoding
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix, classifi
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_sc
```

In [8]:
```python
# Encode 'type' column (TV Show / Movie) into numeric
netflix_df_encoded = netflix_df.copy()
le = LabelEncoder()
netflix_df_encoded['type_encoded'] = le.fit_transform(netflix_df_encoded['type']
```

In [9]:
```python
# Features and target
X = netflix_df_encoded[['release_year', 'type_encoded']]
y = netflix_df_encoded['influenced']

print("X")
print(X)
print()
print("y")
print(y)
```

```
X
      release_year  type_encoded
2             2021             1
7             1993             0
18            2021             0
19            2021             1
24            1998             0
...            ...           ...
8801          2015             0
8802          2007             0
8804          2009             0
8805          2006             0
8806          2015             0

[1630 rows x 2 columns]

y
2       0
7       0
18      0
19      0
24      0
       ..
8801    0
8802    0
8804    0
8805    0
8806    0
Name: influenced, Length: 1630, dtype: int64
```

In [10]:
```
# Step 2: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratif


print(X_train)
print()
print(X_test)
print()
print(y_train)
print()
print(y_test)
```

```
      release_year  type_encoded
890           2021             1
8651          2008             0
6098          2012             0
4097          2018             0
4587          2011             0
...            ...           ...
6616          2017             0
4273          2018             1
4094          2018             0
5778          2015             0
5031          2017             0

[1304 rows x 2 columns]

      release_year  type_encoded
7379          2016             0
8080          2015             0
5552          2016             0
5584          2016             0
4129          2018             1
...            ...           ...
2792          2020             1
1896          2020             0
7503          2005             0
2122          2020             1
7169          2008             0

[326 rows x 2 columns]

890     0
8651    1
6098    0
4097    0
4587    0
       ..
6616    0
4273    0
4094    0
5778    0
5031    0
Name: influenced, Length: 1304, dtype: int64

7379    1
8080    0
5552    0
5584    0
4129    0
       ..
2792    0
1896    0
7503    0
2122    0
7169    0
Name: influenced, Length: 326, dtype: int64
```

In [11]:
```python
# Step 1: Train Random Forest
rf = RandomForestClassifier(
    n_estimators=100,
    class_weight='balanced',   # Handle imbalance
```

```
    random_state=42
)

rf.fit(X_train, y_train)

# Step 2: Predict
y_pred_rf = rf.predict(X_test)

# Step 3: Evaluate
print("📌 Random Forest Classifier Results")
print("Accuracy:", accuracy_score(y_test, y_pred_rf))
print("F1 Score:", f1_score(y_test, y_pred_rf))
print("Precision:", precision_score(y_test, y_pred_rf))
print("Recall:", recall_score(y_test, y_pred_rf))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_rf))
print("\nDetailed Classification Report:\n", classification_report(y_test, y_pre
```

```
📌 Random Forest Classifier Results
Accuracy: 0.8251533742331288
F1 Score: 0.14925373134328357
Precision: 0.12195121951219512
Recall: 0.19230769230769232
Confusion Matrix:
 [[264  36]
 [ 21   5]]

Detailed Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.88      0.90       300
           1       0.12      0.19      0.15        26

    accuracy                           0.83       326
   macro avg       0.52      0.54      0.53       326
weighted avg       0.86      0.83      0.84       326
```

In [12]:
```
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_rep
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Train model
gbc = GradientBoostingClassifier(random_state=42)
gbc.fit(X_train, y_train)
y_pred_gbc = gbc.predict(X_test)
```

In [13]:
```
# Accuracy
print("Accuracy:", accuracy_score(y_test, y_pred_gbc))

# Confusion matrix
cm = confusion_matrix(y_test, y_pred_gbc)
print("Confusion Matrix:\n", cm)

# Classification report
print("\nClassification Report:\n", classification_report(y_test, y_pred_gbc))
```

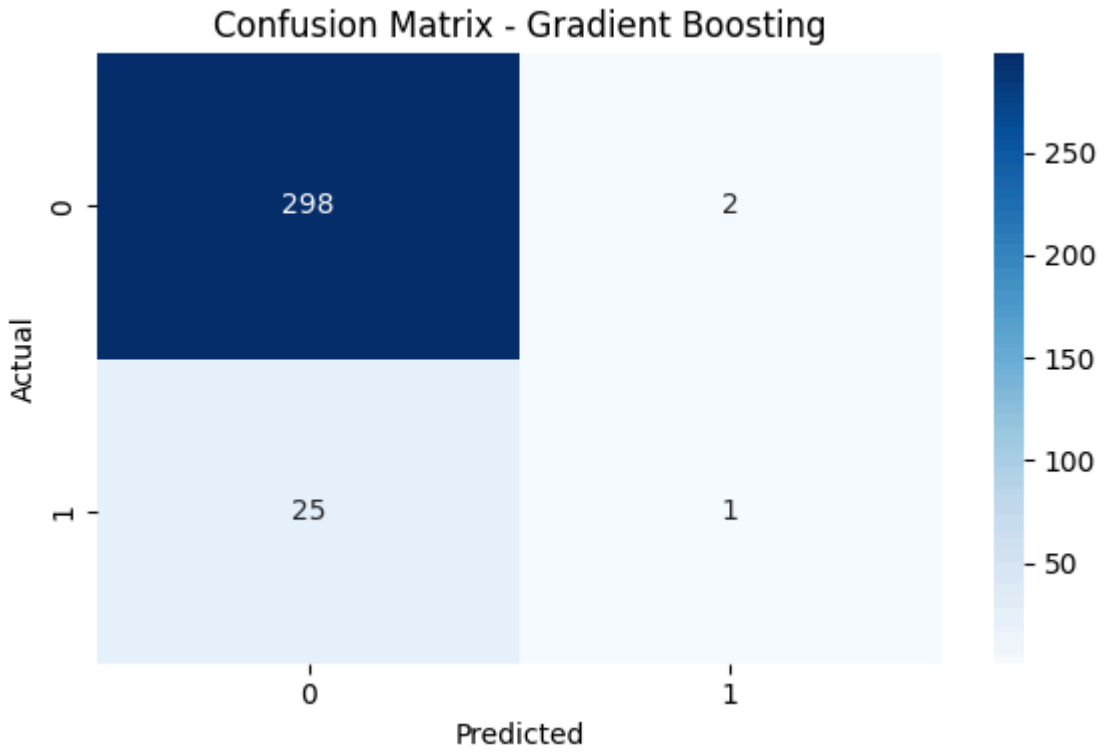
```
Accuracy: 0.9171779141104295
Confusion Matrix:
 [[298   2]
 [ 25   1]]

Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.99      0.96       300
           1       0.33      0.04      0.07        26

    accuracy                           0.92       326
   macro avg       0.63      0.52      0.51       326
weighted avg       0.88      0.92      0.89       326
```

In [14]:
```
plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=[0, 1], yticklabe
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix - Gradient Boosting")
plt.tight_layout()
plt.show()
```



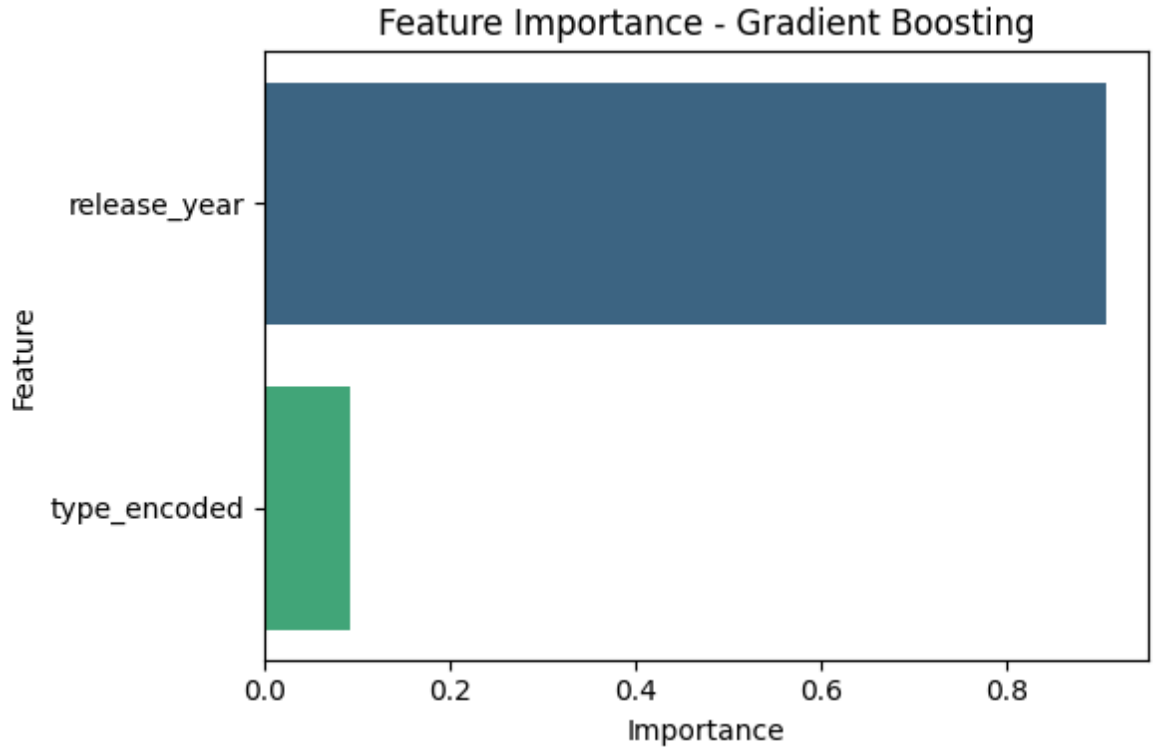
In [15]:
```
importance_df = pd.DataFrame({
    'Feature': X_train.columns,
    'Importance': gbc.feature_importances_
}).sort_values(by='Importance', ascending=False)

plt.figure(figsize=(6, 4))
sns.barplot(x='Importance', y='Feature', data=importance_df, palette='viridis')
plt.title("Feature Importance - Gradient Boosting")
plt.tight_layout()
plt.show()
```

```
C:\Users\himav\AppData\Local\Temp\ipykernel_35260\1604531868.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.barplot(x='Importance', y='Feature', data=importance_df, palette='viridis')
```



In [ ]:

In [ ]: