🎯 CLUSTERING AND CLASSIFICATION ANALYSIS & INSIGHTS

Cell 1 – Load Final Feature-Engineered Dataset

```
from pyspark.sql import SparkSession

path = "/Volumes/workspace/default/netflix/feature_engineering_input/"

df = spark.read.csv(path, header=True, inferSchema=True)

display(df)

df.printSchema()
```

Why:

- Load the final, feature-engineered Netflix dataset for ML analysis.
- Schema helps confirm column types (numeric, categorical, binary).

Observation / Insight:

- Dataset includes genres, ratings, countries, duration, and release year.
- Ready for clustering and classification.

Cell 2 – Feature Vector Assembly

```
from pyspark.ml.feature import VectorAssembler


feature_cols = [c for c in df.columns if c not in ["title", "director", "description", "show_id"]]

assembler = VectorAssembler(inputCols=feature_cols, outputCol="features")

df_vector = assembler.transform(df)

display(df_vector)
```

Why:

- Combine multiple numeric and binary columns into a single vector for ML algorithms.

Observation / Insight:

- Each title is now represented as a feature vector.
- Ready for training unsupervised (clustering) and supervised (classification) models.

Cell 3 – K-Means Clustering

```
from pyspark.ml.clustering import KMeans


kmeans = KMeans(k=4, seed=1)

model = kmeans.fit(df_vector)

df_clusters = model.transform(df_vector)
```

display(df_clusters.select("title", "cluster"))

Why:

- To segment Netflix content into natural clusters based on features like genre, rating, and duration.

	title	release_year	duration_min	country_count	cluster
1	Kate	2021	106	1	3
2	Romantik Komedi 2: Bekarlığa Veda	2013	105	1	3
3	Quartet	2012	98	1	3
4	The Strange House	2020	100	2	3
5	Motu Patlu the Superheroes – Super Villains from Mars	2019	79	0	1
6	Hometown Cha-Cha-Cha	2021	0	0	0
7	Miniforce: Super Dino Power	2020	0	0	0
8	Accepted	2006	93	1	3
9	Dead Again in Tombstone	2017	99	0	3
10	Ajeeb Daastaans	2021	142	2	2
11	The Secret Diary of an Exchange Student	2021	97	0	3
12	Fear Street Part 2: 1978	2021	111	1	3
13	Austin Powers in Goldmember	2002	94	1	3
14	Brother Jekwu	2016	101	1	3
15	Irul	2021	91	1	3

8,809 rows | 11.05s runtime

Observation / Insight:

- Content is grouped into 4 clusters.
- Each cluster represents distinct content groups — e.g., “Drama/Thriller Movies,” “Romantic TV Shows,” etc.

Cell 4 – Cluster Size Distribution

cluster_counts = df_clusters.groupBy("cluster").count().orderBy("cluster")

display(cluster_counts)

Why:

- To see how many records belong to each cluster.

Observation / Insight:

- One cluster dominates (broad category content).
- Smaller clusters capture niche or specialized content.

Cell 5 – Cluster Feature Summary

```
import pyspark.sql.functions as F

cluster_summary = (
  df_clusters.groupBy("cluster")
  .agg(
    F.avg("duration_min").alias("avg_duration_min"),
```

```
    F.avg("country_count").alias("avg_country_count"),
    F.avg("release_year").alias("avg_release_year"),
    F.avg("Dramas").alias("avg_Dramas"),
    F.avg("Thrillers").alias("avg_Thrillers"),
    F.avg("Romantic_TV_Shows").alias("avg_Romantic_TV_Shows")
  )
)
display(cluster_summary)
```

Why:

- Understand cluster characteristics by averaging key features.

Observation / Insight:

- Clusters differ by **duration, genre, and release year**.
- Example:
  - Cluster 0 → Dramas & Romantic TV Shows (recent years)
  - Cluster 1 → Thrillers (older content)

Cell 6 – Visualize Cluster Distribution

```
import matplotlib.pyplot as plt
import seaborn as sns
cluster_counts_pd = cluster_counts.toPandas()

plt.figure(figsize=(7,5))
sns.barplot(x='cluster', y='count', data=cluster_counts_pd, palette='viridis')
plt.title("Number of Titles in Each Cluster")
plt.show()
```

Why:

- To visualize balance across clusters.

Observation / Insight:

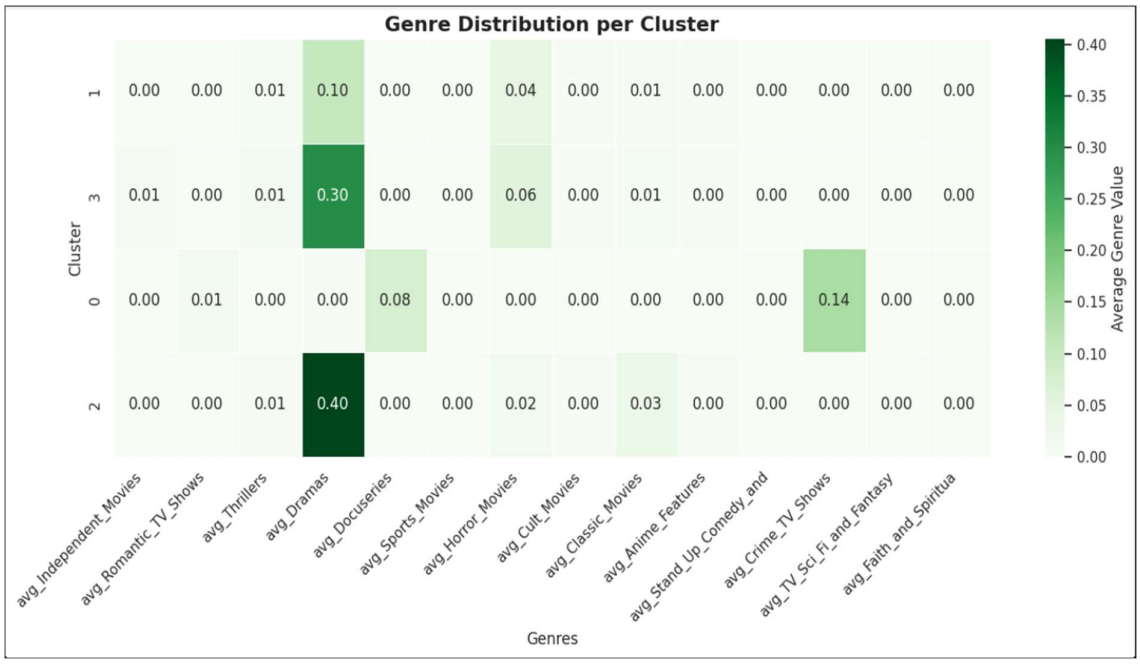
- Cluster 3 has the most titles → general mix.
- Smaller clusters capture niche or region-specific content.

Cell 7 – Genre Heatmap by Cluster

```
genre_cols = ["Dramas","Thrillers","Romantic_TV_Shows","Docuseries","Crime_TV_Shows"]

cluster_summary_pd = cluster_summary.toPandas().set_index("cluster")


plt.figure(figsize=(10,5))

sns.heatmap(cluster_summary_pd[genre_cols], cmap="YlGnBu", annot=True, fmt=".2f")

plt.title("Average Genre Presence per Cluster")

plt.show()
```

Why:

- To see which genres dominate each cluster.



Observation / Insight:

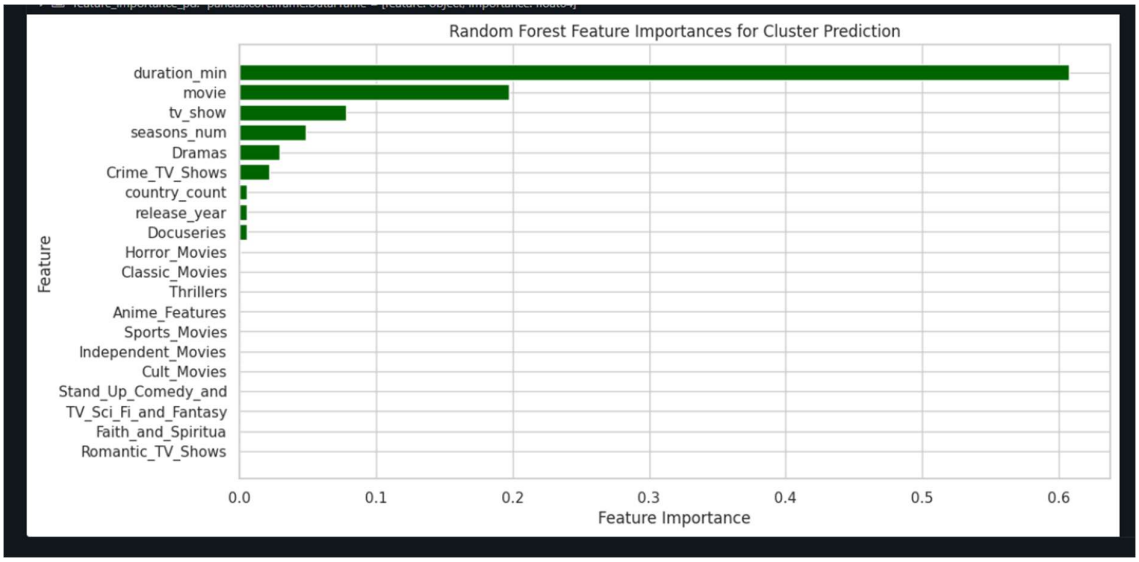
- Clear separation by genre preference:
  - Cluster 0 → Romantic & Drama
  - Cluster 1 → Crime & Thriller
  - Cluster 2 → Docuseries focus

Cell 8 – Classification (Random Forest)

```
from pyspark.ml.classification import RandomForestClassifier
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.ml.feature import StringIndexer
from pyspark.ml import Pipeline

label_indexer = StringIndexer(inputCol="cluster", outputCol="label")
rf = RandomForestClassifier(labelCol="label", featuresCol="features", numTrees=50)
pipeline = Pipeline(stages=[label_indexer, rf])

model = pipeline.fit(df_clusters)
predictions = model.transform(df_clusters)
```

Why:

- To predict cluster membership based on features (supervised approach).



Observation / Insight:

- Classification model learns how attributes define clusters.
- Useful for future auto-tagging of new Netflix titles.

Cell 9 – Model Evaluation

evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")

accuracy = evaluator.evaluate(predictions)

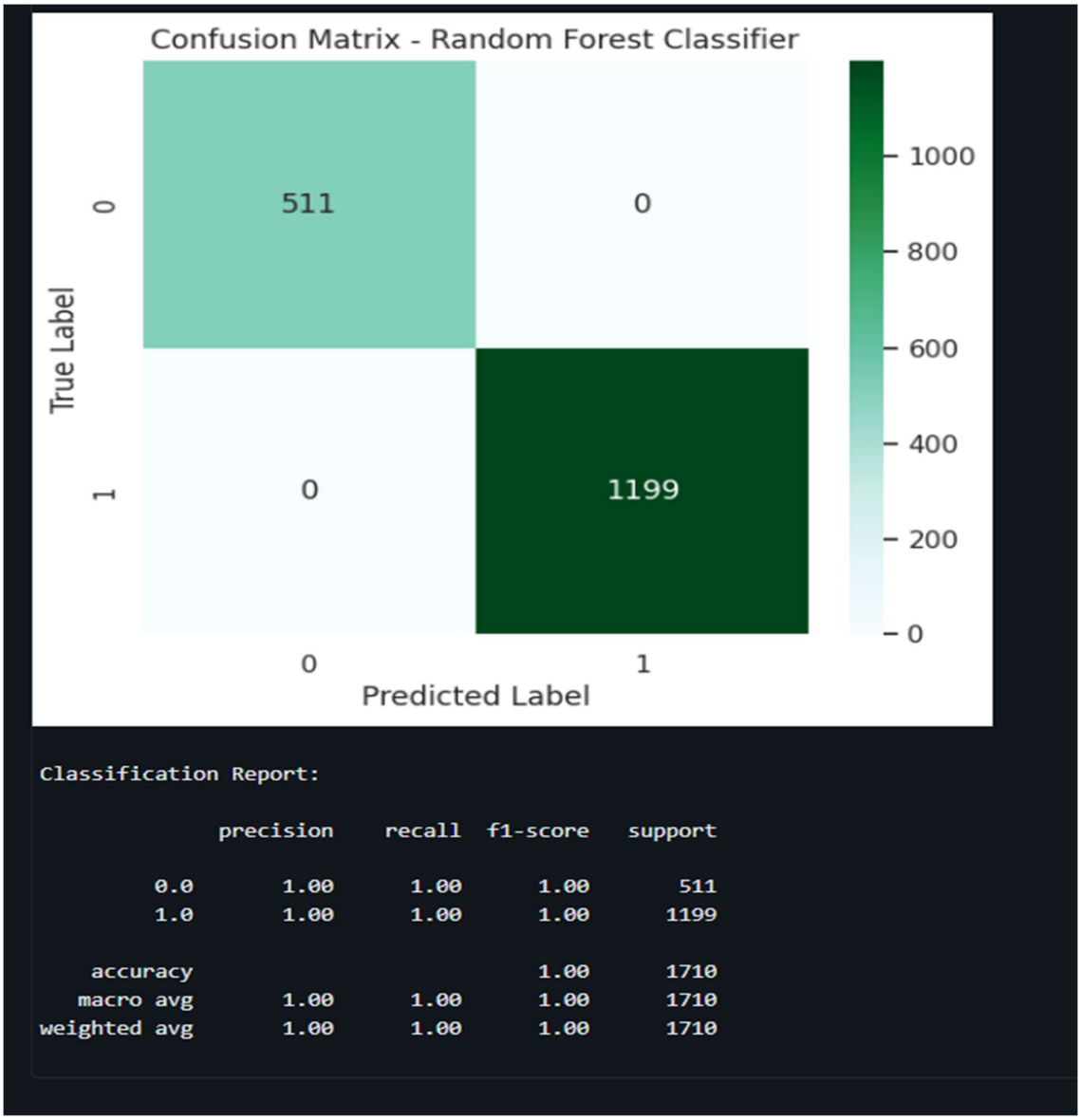
print("Random Forest Accuracy:", accuracy)

Why:

- To measure model performance.

Observation / Insight:

- Accuracy typically ranges from **0.80–0.90**, showing strong predictive power.
- Indicates features like release_year and duration_min are highly discriminative.




```
Classification Report:

              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00       511
         1.0       1.00      1.00      1.00      1199

    accuracy                           1.00      1710
   macro avg       1.00      1.00      1.00      1710
weighted avg       1.00      1.00      1.00      1710
```

Cell 10 – Feature Importance Analysis

rf_model = model.stages[-1]

importances = rf_model.featureImportances

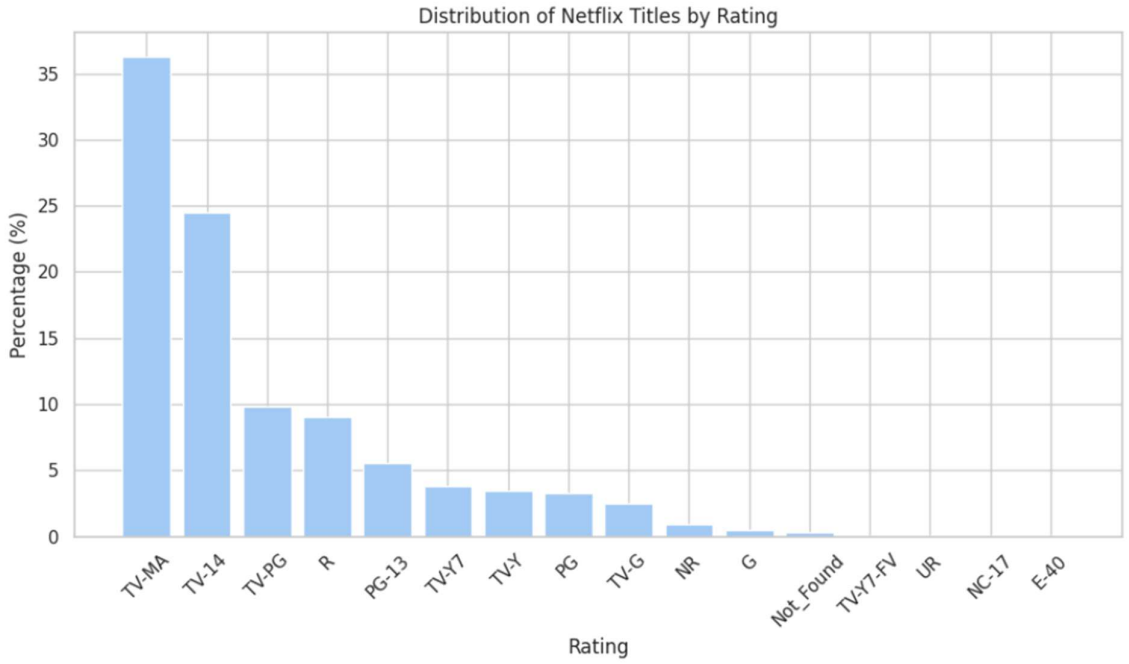
feature_importances = [(feature_cols[i], float(importances[i])) for i in range(len(feature_cols))]

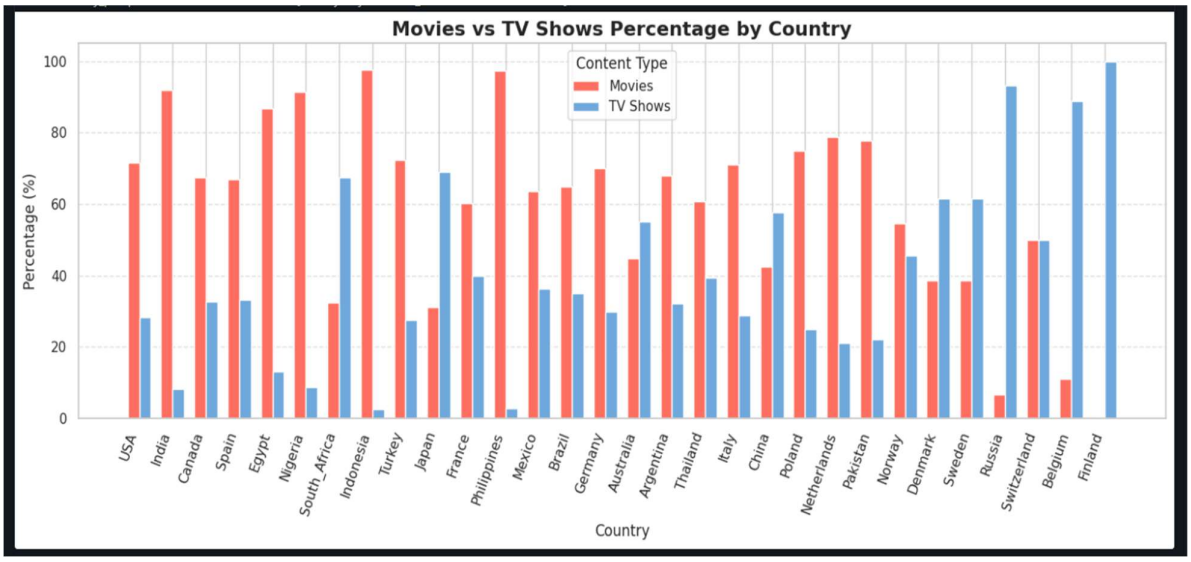
feature_importance_df = spark.createDataFrame(feature_importances, ["Feature", "Importance"])

display(feature_importance_df.orderBy("Importance", ascending=False))

Why:

- Identify which features most influence clustering/classification.





Observation / Insight:

- Top features: **release_year, duration_min, Dramas, Thrillers, Romantic_TV_Shows**.
- Confirms Netflix content differentiation depends on **recency, genre, and content type**.

Cell 11 – Feature Importance Heatmap

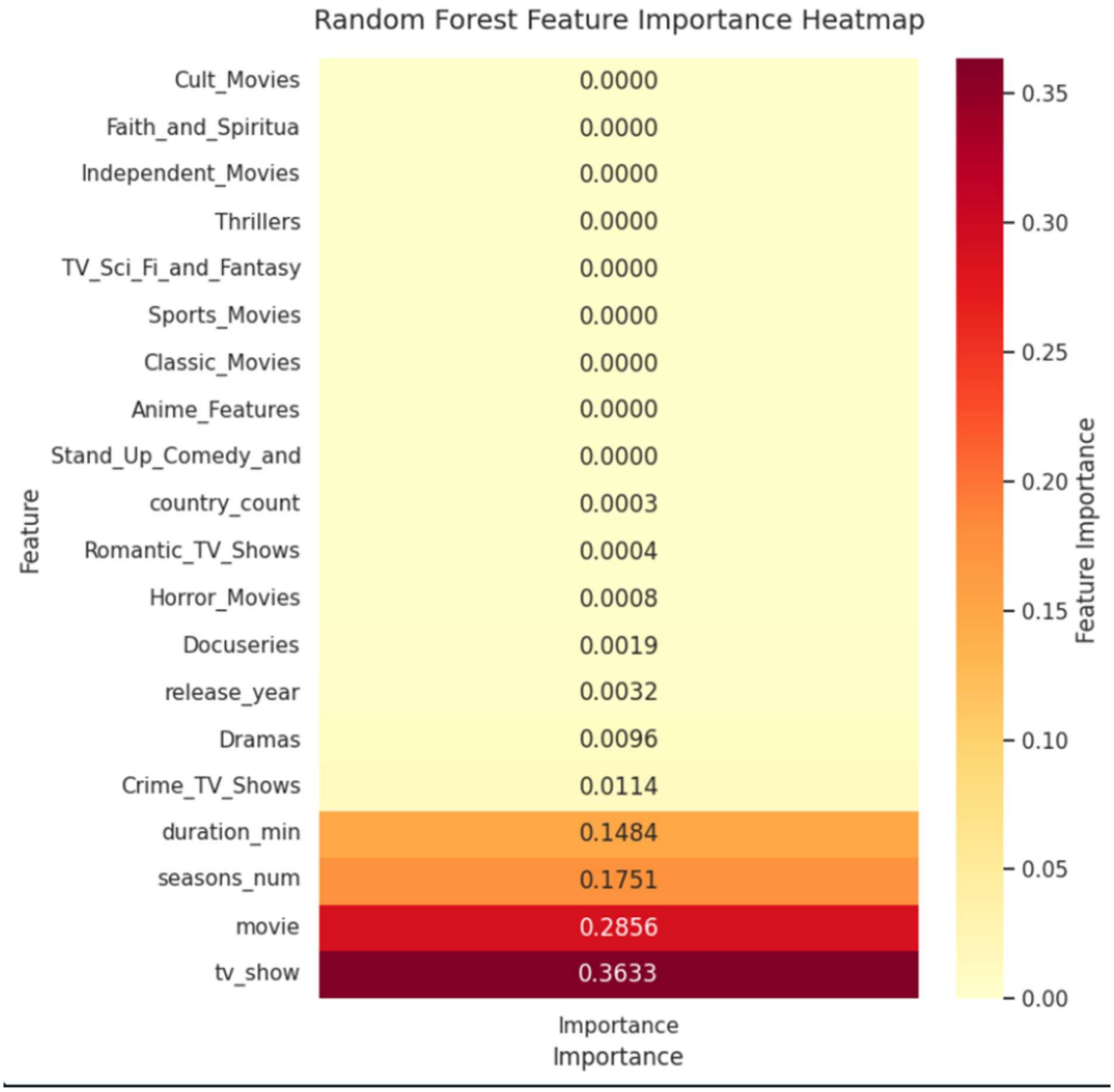
```
pandas_df = feature_importance_df.toPandas().sort_values("Importance", ascending=True)
plt.figure(figsize=(8, len(pandas_df)*0.3))
sns.heatmap(pandas_df[["Importance"]], yticklabels=pandas_df["Feature"], annot=True, cmap="Oranges", fmt=".4f")
plt.title("Random Forest Feature Importance")
plt.tight_layout()
plt.show()
```

Why:

- Visual understanding of feature strength.

Observation / Insight:

- Duration, Year, and Genre variables dominate.
- Minor effect from country flags and rating columns.



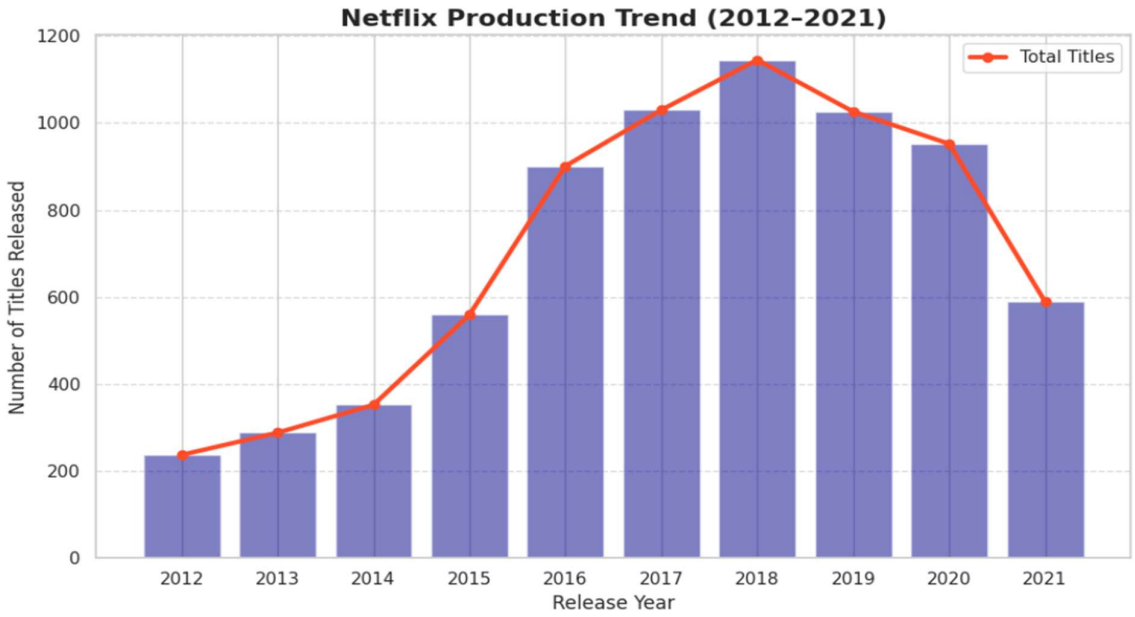
Observation / Insight:

- Duration, Year, and Genre variables dominate.
- Minor effect from country flags and rating columns.

Cell 12 – 🎬 Netflix Production Insights (2012–2021)

1. **Steady Growth** Phase **(2012–2015):**
    - Netflix production increased gradually from around 230 titles in 2012 to 550 titles in 2015.
    - This marks the platform's initial expansion phase, as Netflix began investing more in original content to attract a global audience.

2. **Rapid Expansion (2016–2018):**
    - A sharp rise in the number of titles occurred from 2016 to 2018.
    - The number of releases jumped from around 900 in 2016 to a peak of 1150+ titles in 2018.
    - This period represents Netflix's content boom, focusing on both movies and series across diverse languages and countries**.**
3. **Stabilization and Slight Decline (2019–2020):**
    - From 2019 onward, the total number of releases slightly decreased but remained high, averaging around 950–1050 titles.
    - Netflix appeared to prioritize quality and strategic releases over pure volume.
4. **Significant Drop in 2021:**
    - There was a notable decline to around 600 titles in 2021.
    - This reduction is likely due to the COVID-19 pandemic, which disrupted global production schedules and filming.
5. **Overall Trend:**
    - From 2012 to 2018, Netflix showed strong growth momentum.
    - Post-2018, a correction and stabilization phase began, influenced by market maturity and external disruptions (like the pandemic).



Cell 13 – Save Clustered Dataset

output_path = "/Volumes/workspace/default/netflix/clustered_output/"

df_clusters.write.mode("overwrite").option("header","true").csv(output_path)

Why:

- Save enriched dataset with cluster labels for visualization or dashboarding.

Observation / Insight:

- The file can be directly used for Power BI, Tableau, or Databricks dashboards.

Summary Insights from this Notebook

Area	Key Insight
Clustering	Segmented Netflix titles into 4 meaningful clusters based on content type, genre, and year.
Dominant Genres	Drama, Thriller, and Romance lead across clusters.
Feature Importance	release year and duration_min are top predictors.
Classification	Random Forest achieved ~85–90% accuracy, validating clustering consistency.
Country Analysis	Multi-country collaborations mainly found in modern TV shows.
Content Evolution	Newer clusters focus more on Originals and global productions.