

Ejercicio1

```
EjerciciosPropuestos > Ejercicio1 > Ejercicio1.ipynb > # =====
Generate + Code + Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | O
...
... == EJERCICIO 1: EXPLORACIÓN Y LIMPIEZA DE DATOS - TITANIC ==
PUNTO 1: Archivo train.csv cargado exitosamente con pandas
Dataset cargado: 1,309 filas x 9 columnas
Columnas disponibles: ['Passengerid', 'Age', 'Fare', 'Sex', 'sibsp', 'Parch', 'Pclass', 'Embarked', 'Survived']
PUNTO 2: Primeras 10 filas del dataset:
=====
   Passengerid  Age     Fare  Sex  sibsp  Parch  Pclass  Embarked  Survived
0            1  22.0    7.25   0      1      0      3        2.0       0
1            2  38.0  712.833   1      1      0      1        0.0       1
2            3  26.0    7.925   1      0      0      3        2.0       1
3            4  35.0    53.1    1      1      0      1        2.0       1
4            5  35.0    8.05    0      0      0      3        2.0       0
5            6  28.0   84.583    0      0      0      3        1.0       0
6            7  54.0   518.625   0      0      0      1        2.0       0
7            8  2.0    21.075   0      3      1      3        2.0       0
8            9  27.0   111.333   1      0      2      3        2.0       1
9           10  14.0   300.708   1      1      0      2        0.0       1
PUNTO 3: Valores faltantes por columna:
=====
Passengerid    0
Age            0
Fare           0
...
• Dataset limpio: 1,309 filas x 9 columnas
• Valores nulos en Age: 0 → 0
• Columna Sex convertida a variables dummy usando pd.get_dummies
• Archivo limpio guardado: titanic_limpio.csv
```

```
EjerciciosPropuestos > Ejercicio1 > Ejercicio1_tal_cual.ipynb > # =====
◆ Generate + Code + Markdown | ▶ Run All ▶ Clear All Outputs | ▶ Outline ...
```

... Vista previa de los datos:

	PassengerId	Survived	Pclass	\
0		1	0	3
1		2	1	1
2		3	1	3
3		4	1	1
4		5	0	3

	Name	Sex	Age	SibSp	\
0	Braund, Mr. Owen Harris	male	22.0	1	
1	Cumings, Mrs. John Bradley (Florence Briggs Th... Heikkinen, Miss. Laina	female	38.0	1 0	
2		female	26.0		
3	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	
4	Allen, Mr. William Henry	male	35.0	0	

	Parch	Ticket	Fare	Cabin	Embarked
0	0	A/5 21171	7.2500	NaN	S
1	0	PC 17599	71.2833	C85	C
2	0	STON/O2. 3101282	7.9250	NaN	S
3	0	113803	53.1000	C123	S
4	0	373450	8.0500	NaN	S

Precisión del modelo: 0.80

/tmp/ipykernel_150078/697513106.py:25: FutureWarning: A value is trying to be set
The behavior will change in pandas 3.0. This inplace method will never work because
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
data['Age'].fillna(data['Age'].median(), inplace=True)

Ejercicio2

```
EjerciciosPropuestos > Ejercicio2 > Ejercicio2.ipynb > # =====
Generate + Code + Markdown | Run All Restart Clear All Outputs | Jupyter Variables | O
...
... === EJERCICIO 2: VISUALIZACIÓN DE DATOS - NETFLIX ===

PUNTO 1: Archivo netflix_titles.csv cargado exitosamente
Dataset cargado: 8,807 filas x 12 columnas
Columnas disponibles: ['show_id', 'type', 'title', 'director', 'cast', 'country',
Vista previa de los datos:
   show_id      type            title        director \
0       s1    Movie  Dick Johnson Is Dead  Kirsten Johnson
1       s2  TV Show        Blood & Water           NaN
2       s3  TV Show          Ganglands  Julien Leclercq
3       s4  TV Show     Jailbirds New Orleans           NaN
4       s5  TV Show          Kota Factory           NaN

                                         cast        country \
0                               NaN  United States
1  Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...  South Africa
2  Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...           NaN
3                               NaN           NaN
4  Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...  India

   date_added  release_year rating duration \
0  September 25, 2021      2020  PG-13    90 min
1  September 24, 2021      2021  TV-MA  2 Seasons
2  September 24, 2021      2021  TV-MA   1 Season
...
7. Spain: 232 producciones
8. South Korea: 231 producciones
9. Germany: 226 producciones
10. Mexico: 169 producciones
```

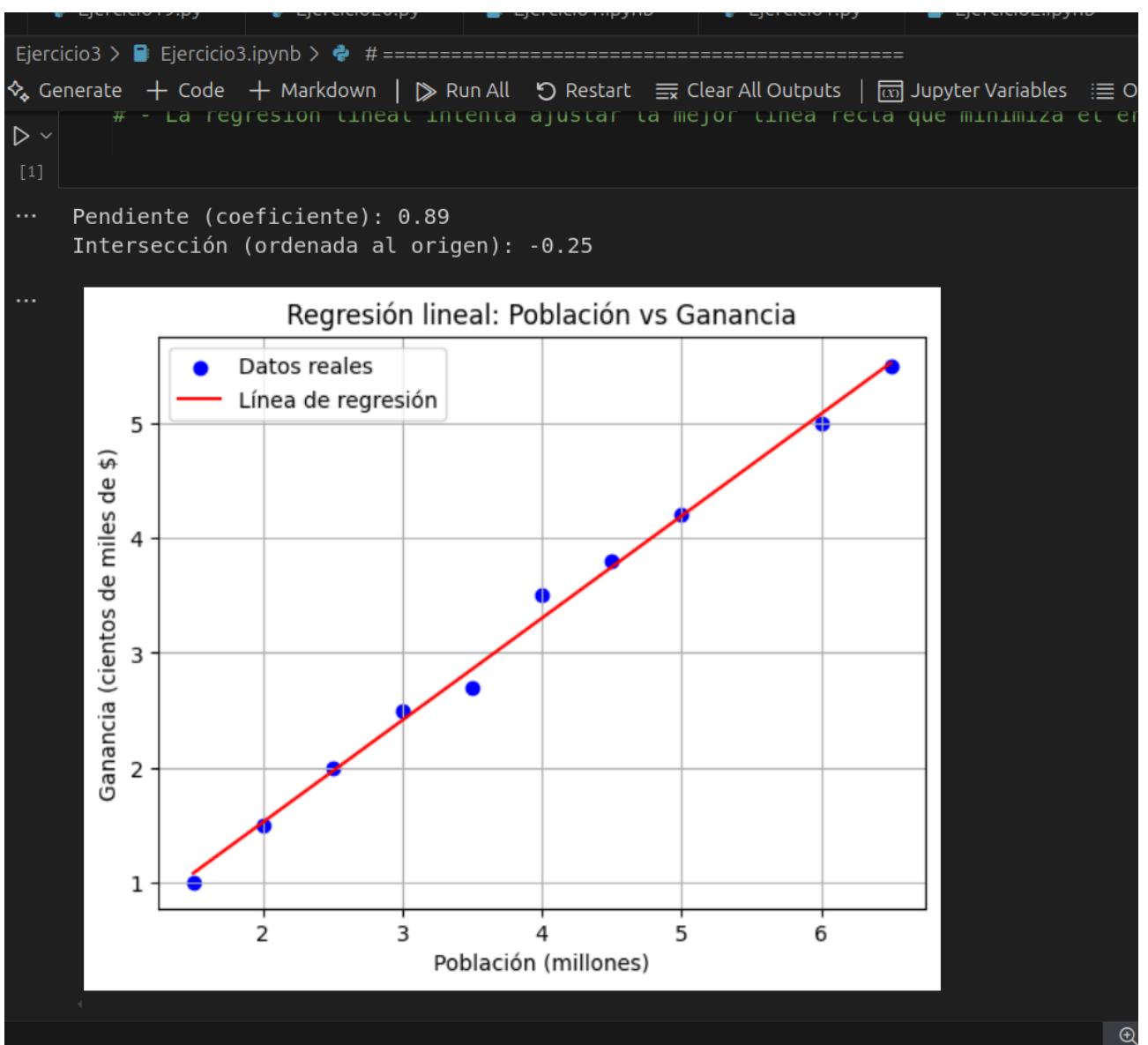
```
EjerciciosPropuestos > Ejercicio2 > Ejercicio2.ipynb > # =====
Generate + Code + Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | O
PUNTO 5: Creando gráfico de torta de directores (top 5)...
=====
TOP 5 DIRECTORES MÁS FRECUENTES:
1. Rajiv Chilaka: 22 producciones
2. Jan Suter: 21 producciones
3. Raúl Campos: 19 producciones
4. Suhas Kadav: 16 producciones
5. Marcus Raboy: 16 producciones
Gráfico de torta guardado como 'netflix_top5_directores_torta.png'

ESTADÍSTICAS ADICIONALES DEL DATASET:
=====
Total de títulos únicos: 8,807
Total de directores únicos: 4,528
Total de países representados: 123
Rango de años: 1925 - 2021

=====
VERIFICACIÓN FINAL DEL CUMPLIMIENTO DEL ENUNCIADO
=====
1. Carga el archivo netflix_titles.csv
2. Muestra cuántos títulos son películas y cuántos son programas de TV
3. Crea un gráfico de barras que muestre cuántas producciones se lanzaron por año
...
- netflix_top10_paises_countplot.png (countplot de países)
- netflix_top5_directores_torta.png (gráfico de torta)

EJERCICIO 2 COMPLETADO EXITOSAMENTE!
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Ejercicio3:



Ejercicio4:

```

.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio4$ python3 Ejercicio4_basic
/home/tunek/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/.venv/lib/python3.12/site-packages/sklearn/linear_model/_logistic.py:187: ConvergenceWarning: lbfgs failed to converge after 200 iteration(s) (status=1):
STOP: TOTAL NO. OF ITERATIONS REACHED LIMIT

Increase the number of iterations to improve the convergence (max_iter=200).
You might also want to scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
    precision      recall   f1-score   support
          0       0.87      0.96      0.91      285
          1       0.86      0.61      0.71      108

   accuracy          0.87      393
  macro avg       0.86      0.79      0.81      393
weighted avg     0.86      0.87      0.86      393

(venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio4$ python3 Ejercicio4.py
== EJERCICIO 4: MODELO DE CLASIFICACIÓN BINARIA - TITANIC SURVIVORS ==

Paso 1: Cargando datos del archivo limpio para ML...
Dataset cargado: 1,309 filas x 9 columnas
Columnas disponibles: ['Passengerid', 'Age', 'Fare', 'sibsp', 'Parch', 'Pclass', 'Embarked', 'Survived', 'Sex_male']

Vista previa del dataset:
   Passengerid  Age     Fare  sibsp  Parch  Pclass  Embarked  Survived  Sex_male
0            1  22.0    7.250     1     0      3        2.0       0         0
1            2  38.0   712.833     1     0      1        0.0       1         1
2            3  26.0    7.925     0     0      3        2.0       1         1
3            4  35.0   53.100     1     0      1        2.0       1         1
4            5  35.0    8.050     0     0      3        2.0       0         0

Tipos de datos:
Passengerid      int64
Age             float64
Fare            float64
sibsp           int64
Parch           int64
Pclass          int64
Embarked        float64
Survived        int64
Sex_male        object

Columna objetivo identificada: 'Survived'

Paso 2: Separando variables...
Variables predictoras ('X'): 8 columnas
Nombres de características: ['Passengerid', 'Age', 'Fare', 'sibsp', 'Parch', 'Pclass', 'Embarked', 'Sex_male']
Variable objetivo ('y'): 'Survived' con distribución:
  No sobrevivió (0): 697 pasajeros (73.9%)
  Sobrevivió (1): 342 pasajeros (26.1%)

Paso 3: Dividiendo datos en entrenamiento y prueba...
Datos de entrenamiento: 916 muestras
Datos de prueba: 393 muestras
Proporción de datos de prueba: 30%

Paso 4: Entrenando modelo de Regresión Logística...
/home/tunek/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/.venv/lib/python3.12/site-packages/sklearn/linear_model/_logistic.py:470: ConvergenceWarning: lbfgs failed to converge after 200 iteration(s) (status=1):
STOP: TOTAL NO. OF ITERATIONS REACHED LIMIT

Increase the number of iterations to improve the convergence (max_iter=200).
You might also want to scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
Modelo entrenado exitosamente

Paso 5: Realizando predicciones...

Paso 6: Evaluando el modelo...
Precisión del modelo: 0.8651 (86.51%)

Reporte de clasificación detallado:
   precision      recall   f1-score   support
No sobrevivió     0.87      0.96      0.91      285
Sobrevivió      0.86      0.61      0.71      108

   accuracy          0.87      393
  macro avg       0.86      0.79      0.81      393
weighted avg     0.86      0.87      0.86      393

```

```

Macro avg    0.86    0.79    0.81    393
weighted avg  0.86    0.87    0.86    393

Matriz de confusión guardada como 'confusion_matrix_titanic_survivors.png'

análisis de importancia de características:

Ranking de características más importantes:
-----
Sex_male      :  2.312 (Aumenta supervivencia)
Pclass        : -0.604 (Disminuye supervivencia)
sibsp         : -0.199 (Disminuye supervivencia)
Embarked      : -0.022 (Disminuye supervivencia)
Age           : -0.012 (Disminuye supervivencia)
Parch         : -0.010 (Disminuye supervivencia)
Passengerid   : -0.003 (Disminuye supervivencia)
Fare          :  0.000 (Aumenta supervivencia)

Ejemplos de predicciones:
real | Predicho | Probabilidad
-----
No sobrevivió | No sobrevivió | 0.016
No sobrevivió | No sobrevivió | 0.056
No sobrevivió | No sobrevivió | 0.024
No sobrevivió | No sobrevivió | 0.059
No sobrevivió | No sobrevivió | 0.129
No sobrevivió | Sobrevivió | 0.611
No sobrevivió | No sobrevivió | 0.153
Sobrevivió | Sobrevivió | 0.559
No sobrevivió | No sobrevivió | 0.191
Sobrevivió | No sobrevivió | 0.196

análisis detallado de la matriz de confusión:
Verdaderos Negativos (TN): 274 - Predijo 'No sobrevivió' y era correcto
Falsos Positivos (FP): 11 - Predijo 'Sobrevivió' pero era incorrecto
Falsos Negativos (FN): 42 - Predijo 'No sobrevivió' pero era incorrecto
Verdaderos Positivos (TP): 66 - Predijo 'Sobrevivió' y era correcto

Métricas adicionales:
Precisión (Precision): 0.8571 - De los que predijo como supervivientes, cuántos eran correctos
Recall (Sensibilidad): 0.6111 - De los supervivientes reales, cuántos detectó
F1-Score: 0.7135 - Media aritmética de precisión y recall

ESTADÍSTICAS FINALES:

```

```

Ejemplos de predicciones:
real | Predicho | Probabilidad
-----
No sobrevivió | No sobrevivió | 0.016
No sobrevivió | No sobrevivió | 0.056
No sobrevivió | No sobrevivió | 0.024
No sobrevivió | No sobrevivió | 0.059
No sobrevivió | No sobrevivió | 0.129
No sobrevivió | Sobrevivió | 0.611
No sobrevivió | No sobrevivió | 0.153
Sobrevivió | Sobrevivió | 0.559
No sobrevivió | No sobrevivió | 0.191
Sobrevivió | No sobrevivió | 0.196

análisis detallado de la matriz de confusión:
Verdaderos Negativos (TN): 274 - Predijo 'No sobrevivió' y era correcto
Falsos Positivos (FP): 11 - Predijo 'Sobrevivió' pero era incorrecto
Falsos Negativos (FN): 42 - Predijo 'No sobrevivió' pero era incorrecto
Verdaderos Positivos (TP): 66 - Predijo 'Sobrevivió' y era correcto

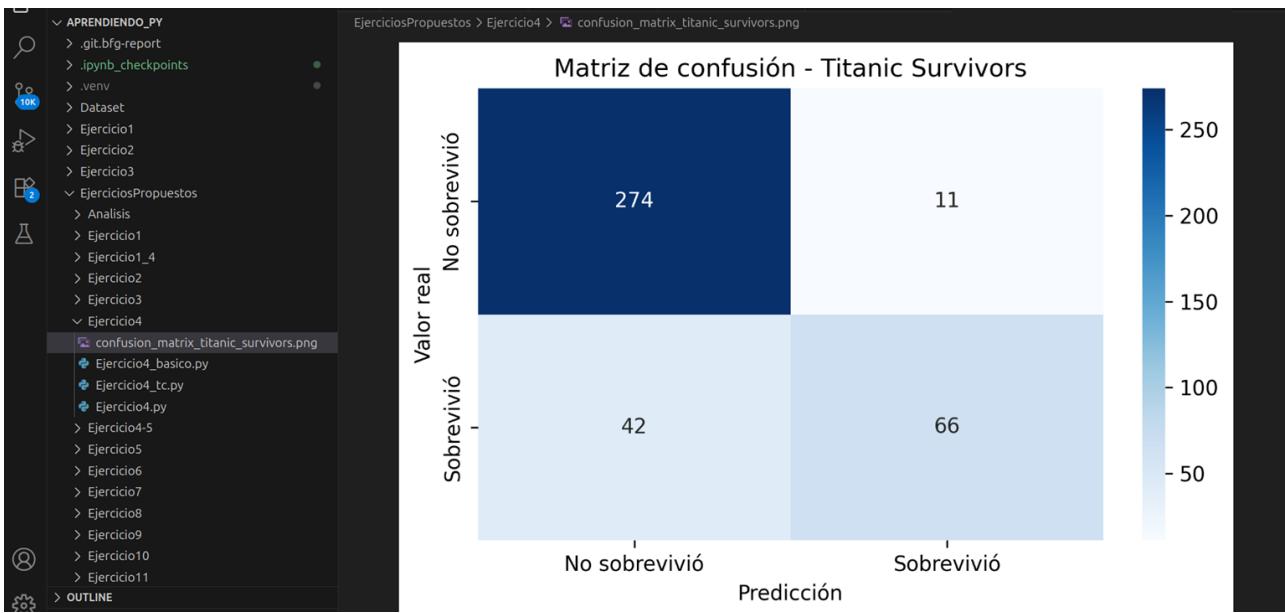
Métricas adicionales:
Precisión (Precision): 0.8571 - De los que predijo como supervivientes, cuántos eran correctos
Recall (Sensibilidad): 0.6111 - De los supervivientes reales, cuántos detectó
F1-Score: 0.7135 - Media aritmética de precisión y recall

ESTADÍSTICAS FINALES:
=====
Total de pasajeros analizados: 393
Predicciones correctas: 340
Predicciones incorrectas: 53
Precisión final: 0.8651

VARIABLES MÁS INFLUYENTES:
-----
3. Sex_male: favorece la supervivencia
3. Pclass: reduce la supervivencia
4. sibsp: reduce la supervivencia

Modelo de clasificación binaria completado exitosamente
archivo generado: confusion_matrix_titanic_survivors.png

```



Ejercicio5:

```
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio5$ python3 Ejercicio5.py
== EJERCICIO 5: CLASIFICACIÓN DE TEXTO CON TF-IDF - FAKE NEWS ==

Paso 1: Cargando datos de noticias procesadas...
Dataset cargado: 44,898 noticias x 2 columnas
Columnas disponibles: ['text', 'label']

Vista previa del dataset:
   text      label
0 Donald Trump just couldn't wish all Americans...      0
1 House Intelligence Committee Chairman Devin Nu...      0
2 On Friday, it was revealed that former Milwauk...      0
3 On Christmas day, Donald Trump announced that ...      0
4 Pope Francis used his annual Christmas Day mes...      0

Distribución de etiquetas:
  Noticias falsas (0): 23,481 noticias (52.3%)
  Noticias verdaderas (1): 21,417 noticias (47.7%)

Estadísticas del texto:
  Longitud promedio: 2469 caracteres
  Longitud mínima: 1 caracteres
  Longitud máxima: 51,794 caracteres
  Mediana: 2186 caracteres

Paso 2: Vectorizando texto con TF-IDF...
Mátriz TF-IDF creada:
  Forma: (44898, 5000) (noticias x características)
  Tipo: <class 'scipy.sparse._csr.csr_matrix'> (matriz dispersa)
  Características extraídas: 5,000

Ejemplos de características (palabras/n-gramas):
  1. '00'
  2. '00 pm'
  3. '000'
  4. '000 people'
  5. '10'
  6. '10 000'
  7. '10 percent'
  8. '10 years'
  9. '100'
```

```

12. '12'
13. '120'
14. '13'
15. '14'
16. '15'
17. '150'
18. '16'
19. '17'
20. '18'

Paso 3: Dividiendo datos en entrenamiento y prueba...
Datos de entrenamiento: 31,428 noticias
Datos de prueba: 13,470 noticias
Proporción de datos de prueba: 30%

Distribución en conjunto de entrenamiento:
Falsas: 16,436 (52.3%)
Verdaderas: 14,992 (47.7%)

Paso 4: Entrenando modelo Naive Bayes...
Modelo Multinomial Naive Bayes entrenado exitosamente

Paso 5: Realizando predicciones...

Paso 6: Evaluando el modelo...
Precisión del modelo: 0.9458 (94.50%)

Reporte de clasificación detallado:
      precision    recall   f1-score   support
Noticia falsa       0.95      0.95      0.95     7045
Noticia verdadera   0.94      0.94      0.94     6425

   accuracy         0.94      13470
  macro avg        0.94      0.94      0.94     13470
weighted avg       0.95      0.94      0.94     13470

Matriz de confusión guardada como 'confusion_matrix_fake_news.png'

Análisis de palabras más importantes para cada clase:

```

```

accuracy         0.94      13470
macro avg        0.94      0.94      0.94     13470
weighted avg     0.95      0.94      0.94     13470

Matriz de confusión guardada como 'confusion_matrix_fake_news.png'

Análisis de palabras más importantes para cada clase:

Top 10 palabras indicativas de NOTICIAS FALSAS:
1. 'trump' (score: -4.777)
2. 'clinton' (score: -5.784)
3. 'people' (score: -5.866)
4. 'obama' (score: -5.893)
5. 'president' (score: -5.904)
6. 'hillary' (score: -5.947)
7. 'just' (score: -5.954)
8. 'said' (score: -5.976)
9. 'like' (score: -6.051)
10. 'donald' (score: -6.159)

Top 10 palabras indicativas de NOTICIAS VERDADERAS:
1. 'said' (score: -4.821)
2. 'trump' (score: -5.237)
3. 'reuters' (score: -5.344)
4. 'president' (score: -5.773)
5. 'state' (score: -5.929)
6. 'government' (score: -5.968)
7. 'house' (score: -6.021)
8. 'republican' (score: -6.112)
9. 'washington' (score: -6.121)
10. 'united' (score: -6.146)

Ejemplos de predicciones:
Real | Predicho | Probabilidad | Texto (primeros 80 caracteres)
-----
Falsa | Falsa | 0.029 | This news should be enough to end Hillary s obsession with becoming our next Pre...
Verdadera | Verdadera | 0.992 | MONROVIA (Reuters) - Liberia s President Ellen Johnson Sirleaf on Tuesday said d...
Falsa | Falsa | 0.001 | Trump campaign manager Kellyanne Conway ran into a buzzsaw named Wolf Blitzer on...
Verdadera | Verdadera | 0.995 | MOSCOW (Reuters) - It is hard to hold a dialogue with the current U.S. administr...
Falsa | Falsa | 0.477 | ...

```

```

4. 'president' (score: -5.773)
5. 'state' (score: -5.929)
6. 'government' (score: -5.968)
7. 'house' (score: -6.021)
8. 'republican' (score: -6.112)
9. 'washington' (score: -6.121)
10. 'united' (score: -6.146)

Ejemplos de predicciones:
Real | Predicho | Probabilidad | Texto (primeros 80 caracteres)
-----
Falsa | Falsa | 0.029 | This news should be enough to end Hillary's obsession with becoming our next Pre...
Verdadera | Verdadera | 0.992 | MONROVIA (Reuters) - Liberia's President Ellen Johnson Sirleaf on Tuesday said d...
Falsa | Falsa | 0.001 | Trump campaign manager Kellyanne Conway ran into a buzzsaw named Wolf Blitzer on...
Verdadera | Verdadera | 0.995 | MOSCOW (Reuters) - It is hard to hold a dialogue with the current U.S. administr...
Falsa | Falsa | 0.477 | ...

Análisis detallado de la matriz de confusión:
Verdaderos Negativos (TN): 6,663 - Predijo 'Falsa' y era correcto
Falsos Positivos (FP): 382 - Predijo 'Verdadera' pero era falsa
Falsos Negativos (FN): 359 - Predijo 'Falsa' pero era verdadera
Verdaderos Positivos (TP): 6,066 - Predijo 'Verdadera' y era correcto

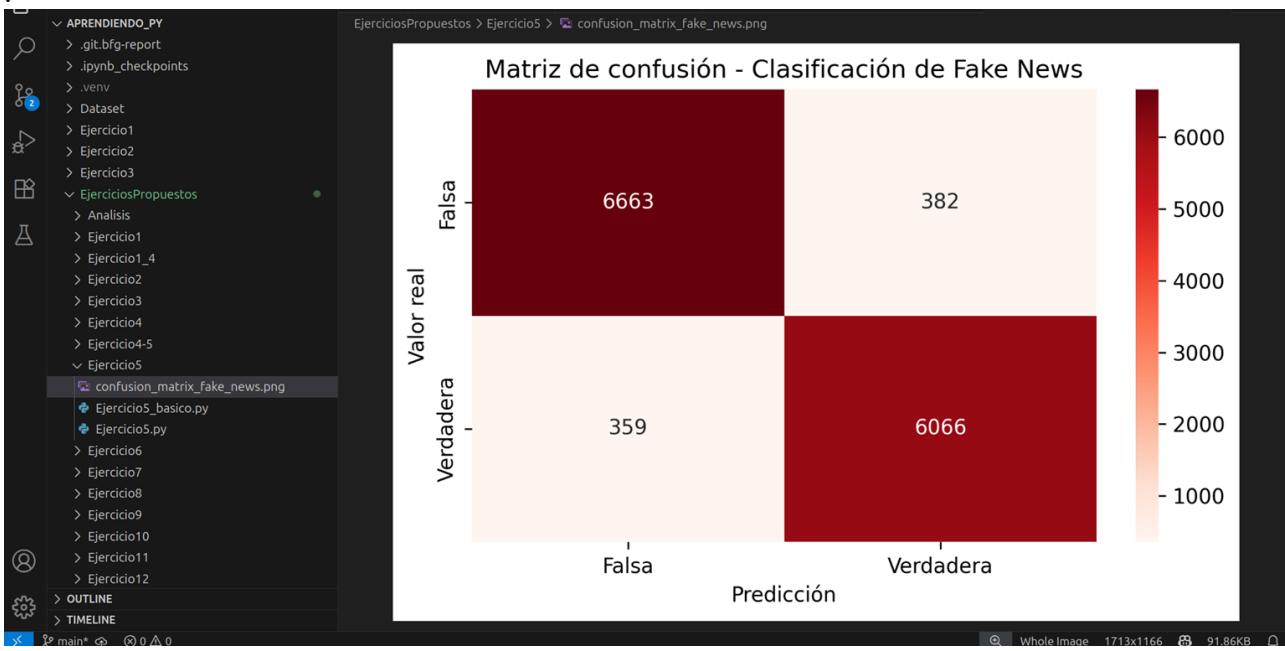
Métricas adicionales:
Precisión: 0.9408 - De las noticias que predijo como verdaderas, cuántas eran correctas
Recall: 0.9441 - De las noticias verdaderas reales, cuántas detectó
F1-Score: 0.9424 - Media aritmética de precisión y recall

ESTADÍSTICAS FINALES:
=====
Total de noticias analizadas: 13,470
Predicciones correctas: 12,729
Predicciones incorrectas: 741
Precisión final: 0.9450

Características del modelo TF-IDF:
Vocabulario total: 5,000 términos
Matriz de características: 44,898 x 5,000
Densidad de la matriz: 0.025778

Modelo de clasificación de texto completado exitosamente
Archivo generado: confusion_matrix_fake_news.png
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio5$ |

```



```

.tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio$ python3 Ejercicio6.py
== EJERCICIO 6: ANALISIS EXPLORATORIO EXTENDIDO - NETFLIX ==

Paso 1: Cargando dataset de Netflix...
Dataset cargado: 8,807 títulos x 12 columnas
Columnas disponibles: ['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added', 'release_year', 'rating', 'duration', 'listed_in', 'description']

Vista previa del dataset:
   show_id      type          title    director ... rating duration                         listed_in                                     description
0     s1  Movie  Dick Johnson Is Dead  Kirsten Johnson ...  PG-13    90 min  Documentaries As her father nears the end of his life, film...
1     s2  TV Show        Blood & Water       NaN ...  TV-MA  2 Seasons  International TV Shows, TV Dramas, TV Mysteries After crossing paths at a party, a Cape Town t...
2     s3  TV Show         Ganglands  Julien Leclercq ...  TV-MA  1 Season  Crime TV Shows, International TV Shows, TV Act...  To protect his family from a powerful drug lor...
3     s4  TV Show  Jailbirds New Orleans       NaN ...  TV-MA  1 Season  Docuseries, Reality TV  Feuds, flirtations and toilet talk go down amo...
4     s5  TV Show        Kota Factory       NaN ...  TV-MA  2 Seasons  International TV Shows, Romantic TV Shows, TV ...  In a city of coaching centers known to train I...

[5 rows x 12 columns]

Información general del dataset:
Memoria utilizada: 7.79 MB
Tipos de datos:
show_id      object
type         object
title        object
director     object
cast          object
country      object
date_added   object
release_year int64
rating        object
duration     object
listed_in    object
description   object
dtype: object

Paso 2: Realizando limpieza básica...
Procesando fechas de adición...
Fechas procesadas:
  Rango de años de adición: 2008 - 2021
  Rango de años de lanzamiento: 1925 - 2021

Valores faltantes por columna:
  director: 2,634 (29.9%)
  cast: 825 (9.4%)

Country: 851 (9.4%)
date_added: 98 (1.1%)
rating: 4 (0.0%)
duration: 3 (0.0%)
year_added: 98 (1.1%)
month_added: 98 (1.1%)

Paso 3: Analizando distribución Movies vs TV Shows por año...
Gráfico guardado como 'netflix_movies_vs_shows_por_año.png'

Estadísticas de contenido por año:
type      Movie  TV Show
year_added
2017.0      839     325
2018.0     1237     388
2019.0     1424     575
2020.0     1284     594
2021.0      993     505

Paso 4: Analizando duración de películas...
Total de películas en el dataset: 6,131
Películas con duración válida: 6,128

Estadísticas de duración de películas:
  Duración promedio: 99.6 minutos
  Duración mediana: 98.0 minutos
  Duración mínima: 3 minutos
  Duración máxima: 312 minutos
  Desviación estándar: 28.3 minutos
Gráfico guardado como 'netflix_duracion_peliculas.png'

Paso 5: Analizando directores más frecuentes...
Total de directores únicos encontrados: 4,993

Top 10 directores más frecuentes:
  1. Rajiv Chilaka: 22 producciones
  2. Jan Suter: 21 producciones
  3. Raúl Campos: 19 producciones
  4. Suhas Kadav: 16 producciones
  5. Marcus Raboy: 16 producciones
  6. Jay Karas: 15 producciones
  7. Cathy Garcia-Molina: 13 producciones
  8. Martin Scorsese: 12 producciones
  9. Michael Moore: 11 producciones
  10. Christopher Guest: 10 producciones

```

```
9. Youssef Chahine: 12 producciones  
10. Jay Chapman: 12 producciones  
Gráfico guardado como 'netflix_top_directores.png'  
Análisis adicional: Paises productores...  
Top 10 paises productores:  
1. United States: 3690 producciones  
2. India: 1046 producciones  
3. United Kingdom: 806 producciones  
4. Canada: 445 producciones  
5. France: 393 producciones  
6. Japan: 318 producciones  
7. Spain: 232 producciones  
8. South Korea: 231 producciones  
9. Germany: 226 producciones  
10. Mexico: 169 producciones  
Gráfico guardado como 'netflix_top_paises.png'
```

Análisis de géneros más populares...

```
Top 15 géneros más frecuentes:  
1. International Movies: 2752 producciones  
2. Dramas: 2427 producciones  
3. Comedies: 1674 producciones  
4. International TV Shows: 1351 producciones  
5. Documentaries: 869 producciones  
6. Action & Adventure: 859 producciones  
7. TV Dramas: 763 producciones  
8. Independent Movies: 756 producciones  
9. Children & Family Movies: 641 producciones  
10. Romantic Movies: 616 producciones  
11. TV Comedies: 581 producciones  
12. Thrillers: 577 producciones  
13. Crime TV Shows: 470 producciones  
14. Kids' TV: 451 producciones  
15. Docuseries: 395 producciones  
Gráfico guardado como 'netflix_top_generos.png'
```

Analisis temporal: Evolución del catálogo...

Mapa de calor guardado como 'netflix_heatmap_temporal.png'

RESUMEN DE INSIGHTS PRINCIPALES:

```
Analisis de generos mas populares...  
Top 15 géneros más frecuentes:  
1. International Movies: 2752 producciones  
2. Dramas: 2427 producciones  
3. Comedies: 1674 producciones  
4. International TV Shows: 1351 producciones  
5. Documentaries: 869 producciones  
6. Action & Adventure: 859 producciones  
7. TV Dramas: 763 producciones  
8. Independent Movies: 756 producciones  
9. Children & Family Movies: 641 producciones  
10. Romantic Movies: 616 producciones  
11. TV Comedies: 581 producciones  
12. Thrillers: 577 producciones  
13. Crime TV Shows: 470 producciones  
14. Kids' TV: 451 producciones  
15. Docuseries: 395 producciones  
Gráfico guardado como 'netflix_top_generos.png'
```

Analisis temporal: Evolución del catálogo...

Mapa de calor guardado como 'netflix_heatmap_temporal.png'

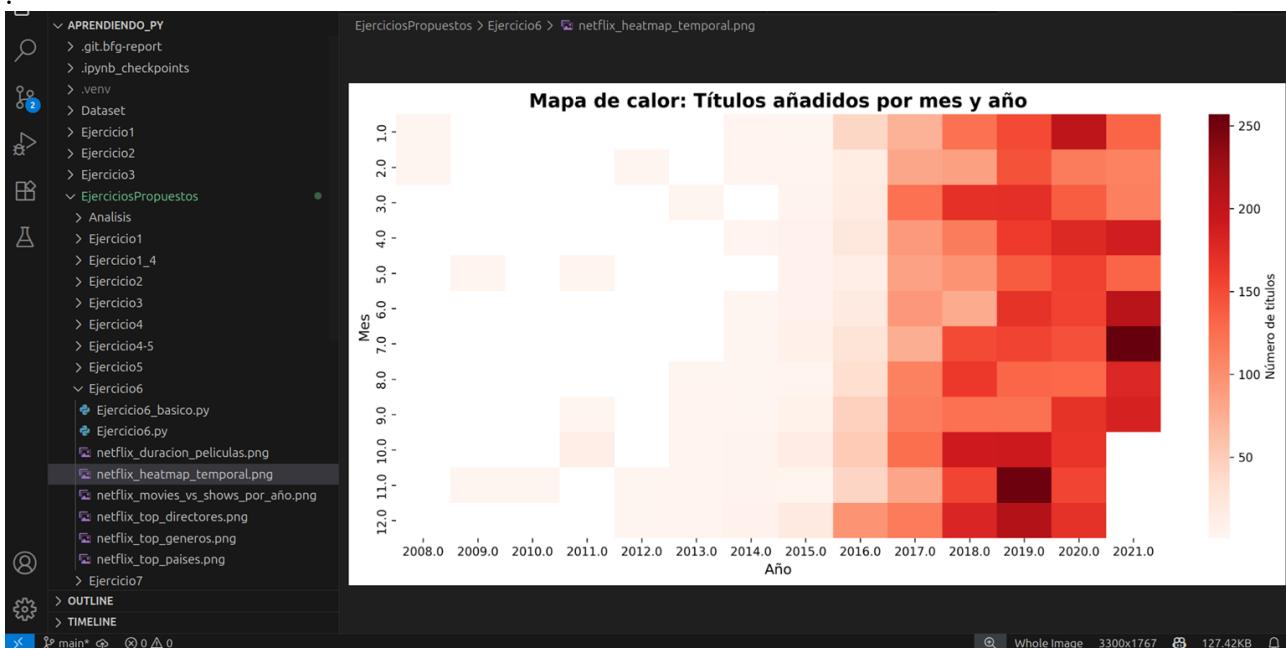
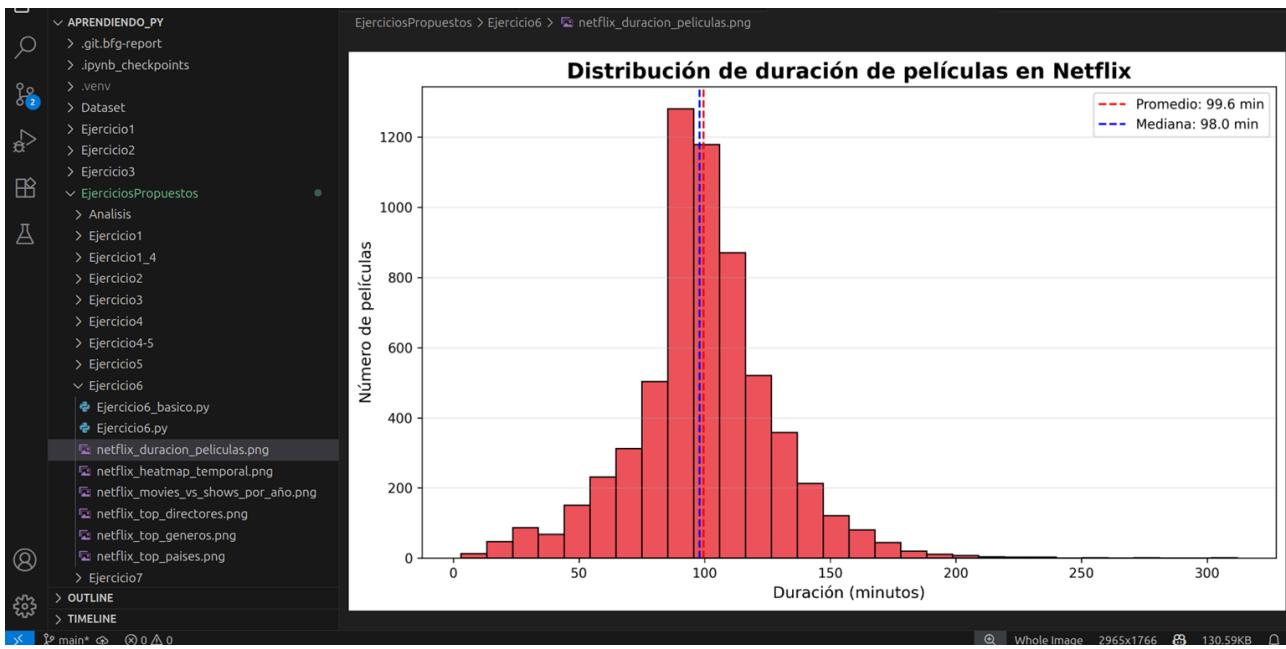
RESUMEN DE INSIGHTS PRINCIPALES:

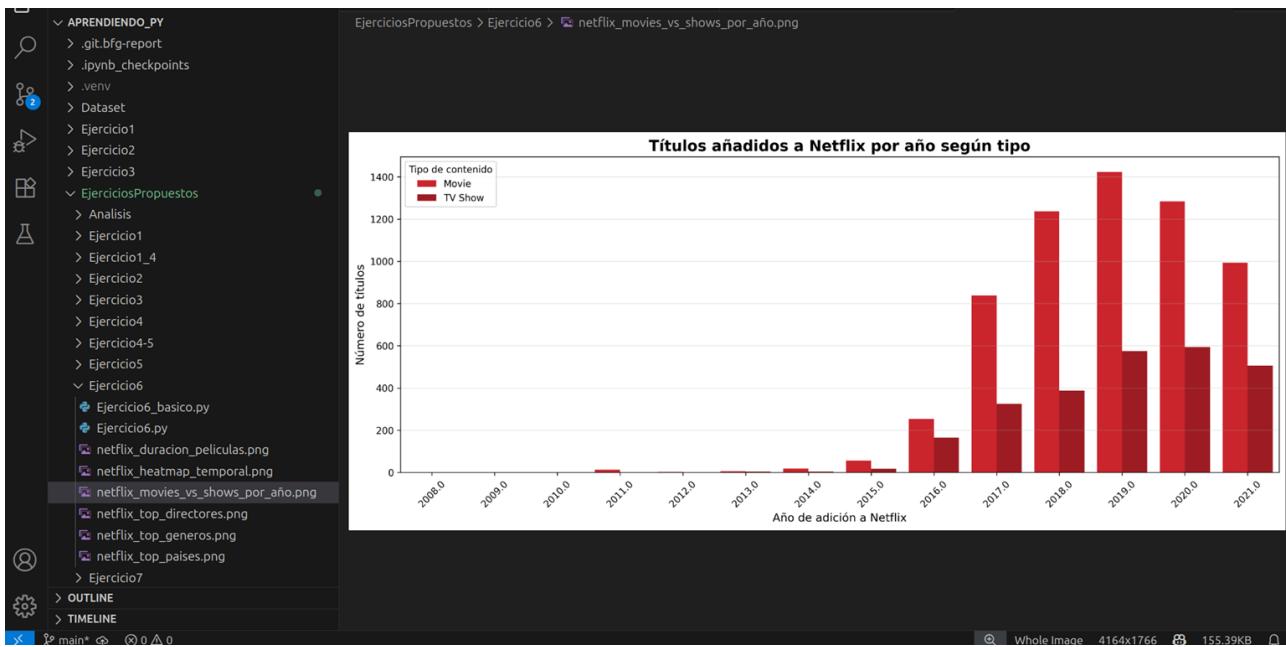
```
=====  
1. CONTENIDO TOTAL: 8,807 títulos en Netflix  
2. DISTRIBUCIÓN: 6,131 películas vs 2,676 series  
3. DIRECTOR TOP: Rajiv Chilaka con 22 producciones  
4. PAÍS TOP: United States con 3690 producciones  
5. GÉNERO TOP: International Movies con 2752 títulos  
6. DURACIÓN PROMEDIO: 100 minutos para películas  
7. AÑO PICO: 2019 (más títulos añadidos)
```

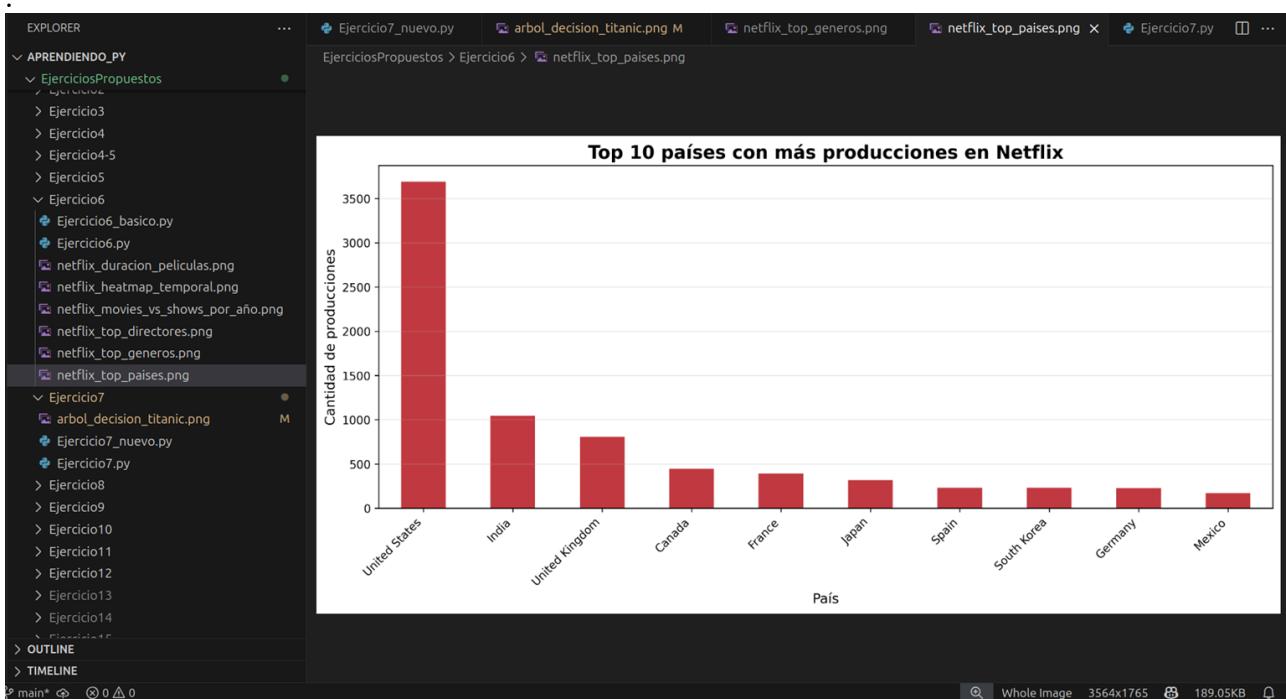
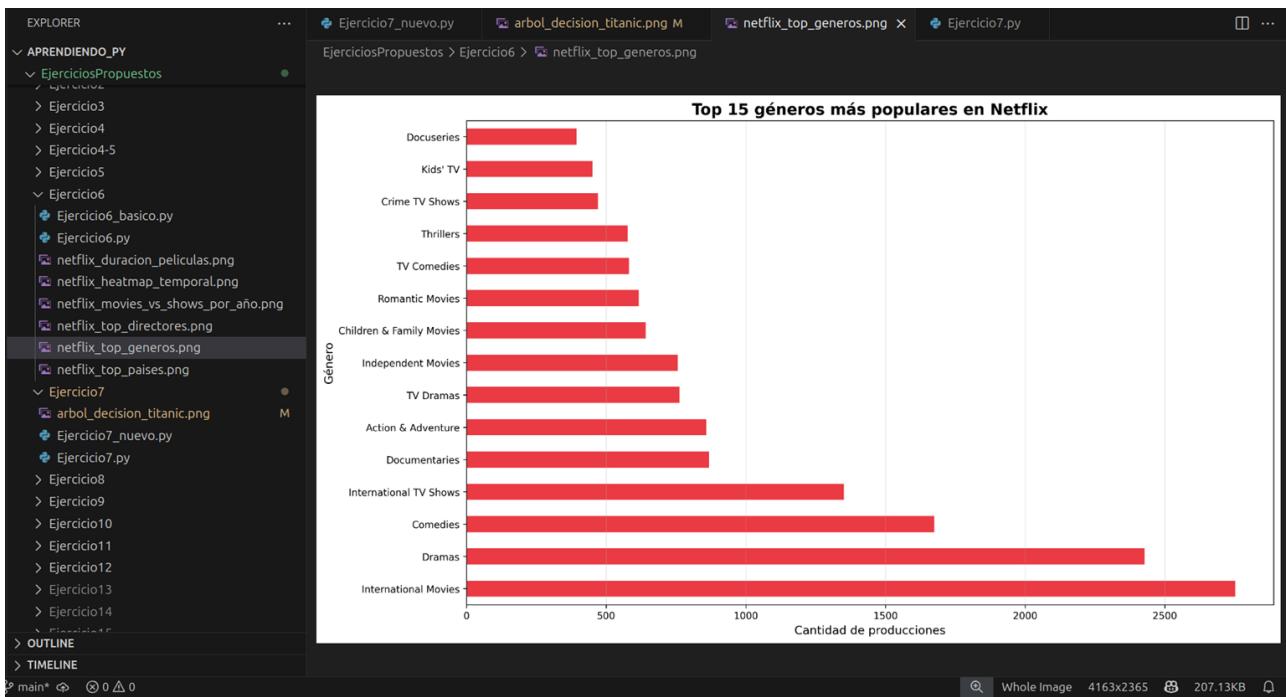
ARCHIVOS GENERADOS:
- netflix_movies_vs_shows_por_año.png
- netflix_duracion_peliculas.png
- netflix_top_directores.png
- netflix_top_paises.png
- netflix_top_generos.png
- netflix_heatmap_temporal.png

EJERCICIO 6: ANÁLISIS EXPLORATORIO COMPLETADO EXITOSAMENTE!
.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicios\$ |

*







Ejercicio 7:

```
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio7$ python3 Ejercicio7.py
== EJERCICIO 7: ÁRBOLES DE DECISIÓN - TITANIC ==

Paso 1: Cargando datos de titanic_limpio.csv...
Dataset cargado: 1309 filas x 9 columnas

Paso 2: Separando variables...
Variables predictoras: 8
Variable objetivo: Survived

Paso 3: Dividiendo datos...
Entrenamiento: 916 muestras
Prueba: 393 muestras

Paso 4: Entrenando árbol de decisión...
Modelo entrenado exitosamente

Paso 5: Visualizando árbol de decisión...
Árbol guardado como 'arbol_decision_titanic.png'

Paso 6: Evaluando modelo...

Reporte de clasificación:
      precision    recall   f1-score  support
          0       0.88     0.92     0.90      285
          1       0.77     0.67     0.71      108

   accuracy                           0.85      393
  macro avg       0.82     0.79     0.81      393
weighted avg       0.85     0.85     0.85      393

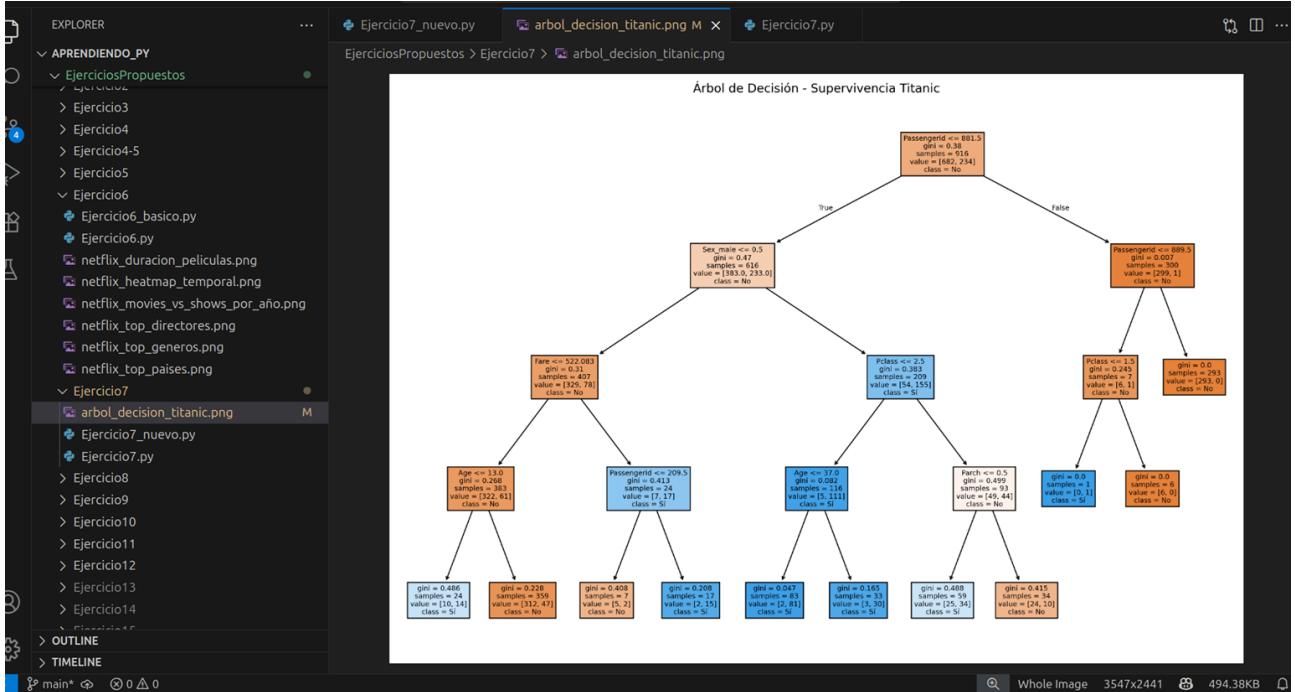
Importancia de características:
 Sex_male: 0.4253
 Passengerid: 0.3082
 Pclass: 0.1317
 Fare: 0.0693
 Age: 0.0480
 Parch: 0.0175
 sibsp: 0.0000
 Embarked: 0.0000

EJERCICIO 7 COMPLETADO
```

```
Ejercitador_nuevo.py Ejercitador.py
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio7$ python3 Ejercicio7_nuevo.py
/home/tunek/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio7/Ejercicio7_nuevo.py:16: UserWarning: Figure
ot be shown
  plt.show()
      precision    recall   f1-score  support
          0       0.88     0.92     0.90      285
          1       0.77     0.67     0.71      108

   accuracy                           0.85      393
  macro avg       0.82     0.79     0.81      393
weighted avg       0.85     0.85     0.85      393

(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio7$ |
```



Ejercicio 8:

```

(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio8$ python3 Ejercicio8.py
== EJERCICIO 8: RANDOM FOREST - FAKE NEWS ===

Paso 1: Cargando datos de noticias_procesadas.csv...
Dataset cargado: 44898 noticias x 2 columnas

Paso 2: Vectorizando texto con TF-IDF...
Matriz TF-IDF: (44898, 5000)
Características extraídas: 5000

Paso 3: Dividiendo datos...
Entrenamiento: 31428 muestras
Prueba: 13470 muestras

Paso 4: Entrenando Random Forest...
Modelo Random Forest entrenado exitosamente

Paso 5: Evaluando modelo...

Reporte de clasificación:
      precision    recall   f1-score   support
          0       1.00     1.00    1.00      7091
          1       1.00     1.00    1.00      6379

   accuracy                           1.00
  macro avg       1.00     1.00    1.00     13470
weighted avg       1.00     1.00    1.00     13470

Top 10 características más importantes:
'reuters': 0.191688
'said': 0.050838
'image': 0.078253
'featured': 0.016984
'just': 0.016397
'washington': 0.013209
'com': 0.013024
'llike': 0.013236
'watch': 0.011560
'read': 0.009224

EJERCICIO 8 COMPLETADO
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio8$ 

```

```

(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio8$ python3 Ejercicio8
Ejercicio8_nuevo.py Ejercicio8.py
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio8$ python3 Ejercicio8_nuevo.py
      precision    recall   f1-score   support
          0       1.00     1.00    1.00      7091
          1       1.00     1.00    1.00      6379

   accuracy                           1.00
  macro avg       1.00     1.00    1.00     13470
weighted avg       1.00     1.00    1.00     13470

(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio8$ 

```

Ejercicio9:

```
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio9$ python3 Ejercicio9.py
== EJERCICIO 9: DETECCIÓN DE ANOMALÍAS - NETFLIX ==
Paso 1: Cargando datos de netflix_titles.csv...
Dataset cargado: 8807 títulos x 12 columnas

Paso 2: Filtrando películas...
Películas encontradas: 6131

Paso 3: Limpiando datos de duración...
Películas con duración válida: 6128
Duración promedio: 99.6 minutos
Duración min-max: 3 - 312 minutos

Paso 4: Detectando anomalías con Isolation Forest...
Modelo Isolation Forest entrenado
Películas normales: 6007
Películas anómalas: 121

Paso 5: Visualizando resultados...
Gráfico guardado como 'anomalias_netflix.png'

Paso 6: Películas con duración anómala:
=====
Silent                                - 3 min
Sol Levante                            - 5 min
Cops and Robbers                      - 8 min
Canvas                                 - 9 min
American Factory: A Conversation with the Obamas - 10 min
Calico Critters: A Town of Dreams      - 11 min
Calico Critters: Everyone's Big Dream Flying in th - 11 min
Besieged Bread                         - 12 min
Zion                                    - 12 min
Cosmos Laundromat: First Cycle       - 12 min
Aziza                                   - 13 min
If Anything Happens I Love You        - 13 min
A StoryBots Space Adventure          - 13 min
Creating The Queen's Gambit           - 14 min
The Road to El Camino: Behind the Scenes of El Cam - 14 min
Buddy Thunderstruck: The Maybe Pile   - 14 min
ANIMA                                  - 15 min
One Like It                            - 15 min
Ya no estoy aquí: Una conversación entre Guillermo - 15 min
Sitarai: Let Girls Dream              - 16 min
The Claudia Kishi Club                - 16 min
=====

```

```
=====
ANIMA                                  - 15 min
One Like It                            - 15 min
Ya no estoy aquí: Una conversación entre Guillermo - 15 min
Sitarai: Let Girls Dream              - 16 min
The Claudia Kishi Club                - 17 min
John Was Trying to Contact Aliens     - 17 min
WHAT DID JACK DO?                     - 17 min
The Battle of Midway                 - 18 min
Pocoyo & Cars                          - 19 min
Michael Lost and Found                - 19 min
Little Miss Sumo                      - 20 min
A Love Song for Latasha              - 20 min
Marked                                 - 21 min
A Go! Go! Cory Carson Summer Camp    - 21 min
Making Nonorthodox                     - 21 min
A Go! Go! Cory Carson Christmas      - 22 min
Lego DC Comics: Batman Be-Leaguered   - 22 min
LEGO Marvel Super Heroes: Black Panther - 22 min
Aggretsuko: We Wish You a Metal Christmas - 22 min
LEGO Marvel Super Heroes: Guardians of the Galaxy - 22 min
A Go! Go! Cory Carson Halloween      - 22 min
LEGO Marvel Spider-Man: Vexed by Venom - 22 min
A Christmas Special: Miraculous: Tales of Ladybug - 22 min
Golden Time                            - 22 min
Ghosts of Sugar Land                  - 22 min
Jake's Buccaneer Blast                - 22 min
Pocoyo Halloween: Spooky Movies       - 22 min
LEGO Marvel Super Heroes: Avengers Reassembled! - 22 min
Pocoyo Carnival                        - 22 min
LEGO: Marvel Super Heroes: Maximum Overload - 22 min
Calico Critters: The Treasure of Calico Village - 22 min
Cloud Atlas                            - 172 min
Saaho                                 - 172 min
Amar Akbar Anthony                   - 172 min
Muqaddar ka Faisla                   - 172 min
My Fair Lady                           - 173 min
Salrat                                - 173 min
Mann                                   - 173 min
Trimurti                               - 173 min
Kaalia                                - 173 min
Sayed the Servant                     - 173 min
Dil Dhadakne Do                       - 174 min
Agneepath                             - 174 min
A Bridge Too Far                      - 176 min
Super Deluxe                           - 176 min
=====
```

together for eternity	- 1/6 min
Hum Saath-Saath Hain	- 176 min
Dil Hai Tumhaara	- 176 min
Bye Bye London	- 177 min
Seven Souls in the Skull Castle 2011	- 177 min
Raja Hindustani	- 177 min
Taal	- 177 min
Manu	- 178 min
Ram Teri Ganga Maili	- 179 min
Happy New Year	- 179 min
The Lord of the Rings: The Two Towers	- 180 min
Blue Is the Warmest Color	- 180 min
Ints Earth of Mankind	- 181 min
Fiddler on the Roof	- 181 min
Jis Desh Men Ganga Behti Hai	- 181 min
Seven Souls in the Skull Castle: Season Flower	- 181 min
Dances with Wolves	- 181 min
King of Boys	- 182 min
Seven Souls in the Skull Castle: Season Bird	- 182 min
Banyuki	- 182 min
Dil Chahta Hai	- 185 min
Seven Souls in the Skull Castle: Season Wind	- 185 min
Swades	- 185 min
Mutiny on the Bounty	- 185 min
Kuch Kuch Hota Hai	- 185 min
Lakshya	- 185 min
Aurora	- 186 min
Kal Ho Naa Ho	- 187 min
Pardes	- 187 min
Magnolia	- 189 min
Unbreakable Kimmy Schmidt: Kimmy vs. the Reverend	- 190 min
The Gospel of Matthew	- 190 min
Wyatt Earp	- 191 min
Maine Pyar Kiya	- 192 min
Kabhi Alvida Naa Kehna	- 192 min
Hum Aapke Hain Koun	- 193 min
Saladin	- 194 min
The Married Couples	- 195 min
Schindler's List	- 195 min
Elephants Dream 4 Hour	- 196 min
Doctor Zhivago	- 200 min
The Lord of the Rings: The Return of the King	- 201 min
What's Your Raashee?	- 203 min
Fifty Year Old Teenager	- 204 min
Seven Souls in the Skull Castle: Season Moon Jogen	- 204 min
The Gospel of Luke	- 205 min

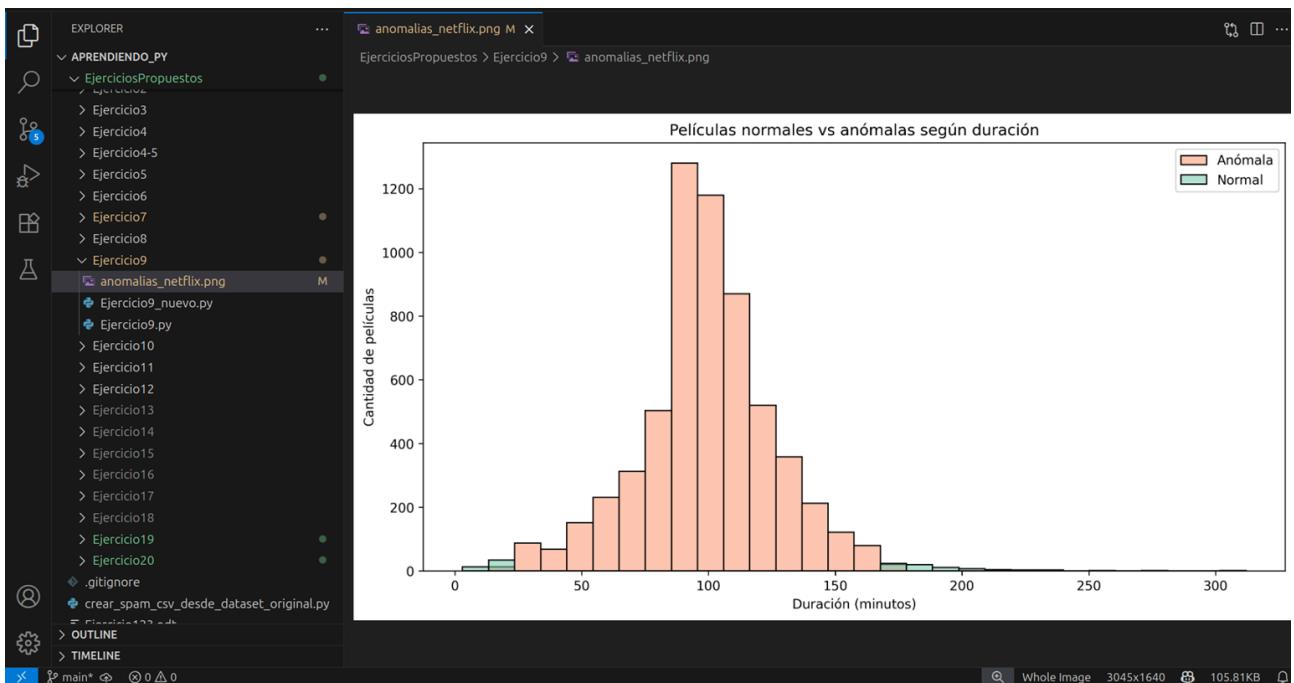
Dil Chahta Hai	- 185 min
Seven Souls in the Skull Castle: Season Wind	- 185 min
Swades	- 185 min
Mutiny on the Bounty	- 185 min
Kuch Kuch Hota Hai	- 185 min
Lakshya	- 185 min
Aurora	- 186 min
Kal Ho Naa Ho	- 187 min
Pardes	- 187 min
Magnolia	- 189 min
Unbreakable Kimmy Schmidt: Kimmy vs. the Reverend	- 190 min
The Gospel of Matthew	- 190 min
Wyatt Earp	- 191 min
Maine Pyar Kiya	- 192 min
Kabhi Alvida Naa Kehna	- 192 min
Hum Aapke Hain Koun	- 193 min
Saladin	- 194 min
The Married Couples	- 195 min
Schindler's List	- 195 min
Elephants Dream 4 Hour	- 196 min
Doctor Zhivago	- 200 min
The Lord of the Rings: The Return of the King	- 201 min
What's Your Raashee?	- 203 min
Fifty Year Old Teenager	- 204 min
Seven Souls in the Skull Castle: Season Moon Jogen	- 204 min
The Gospel of Luke	- 205 min
No Direction Home: Bob Dylan	- 208 min
The Irishman	- 209 min
Kabhi Khushi Kabhi Gham	- 209 min
Seven Souls in the Skull Castle: Season Moon Kagen	- 212 min
Jodhaa Akbar	- 214 min
Lagaan	- 224 min
Sangam	- 228 min
Once Upon a Time in America	- 229 min
Raya and Sakina	- 230 min
Lock Your Girls In	- 233 min
No Longer Kids	- 237 min
The School of Mischief	- 253 min
Headspace: Unwind Your Mind	- 273 min
Black Mirror: Bandersnatch	- 312 min

Rango de duraciones anómalas: 3 - 312 minutos

EJERCICIO 9 COMPLETADO
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio9\$ |

(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio9\$ python3 Ejercicio9_nuevo.py	python3 Ejercicio9_nuevo.py
/home/tunek/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio9/Ejercicio9_nuevo.py:15: UserWarning: FigureCanvasAgg is non-interactive, and thus cann	
ot be shown	
plt.show()	
	title duration
71	A StoryBots Space Adventure 13 min
73	King of Boys 182 min
166	Once Upon a Time in America 229 min
341	Magnolia 189 min
694	Aziza 13 min
...	...
8404	The Lord of the Rings: The Return of the King 201 min
8405	The Lord of the Rings: The Two Towers 179 min
8629	Trifmorti 173 min
8764	Wyatt Earp 191 min
8770	Yaadein 171 min
[119 rows x 2 columns]	

(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio9\$ |



Ejercicio 10:

```
(.venv) tunek@tunek: /Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio10$ python3 Ejercicio10.py
*** EJERCICIO 10: COMPARACIÓN DE MODELOS - TITANIC ***
```

MODELO 1: REGRESIÓN LOGÍSTICA

```
Precisión: 0.8448
precision    recall   f1-score  support
      0       0.85     0.95     0.90      285
      1       0.81     0.57     0.67      188

accuracy                           0.84      393
macro avg       0.83     0.76     0.78      393
weighted avg    0.84     0.84     0.84      393
```

MODELO 2: ÁRBOL DE DECISIÓN

```
Precisión: 0.8524
precision    recall   f1-score  support
      0       0.88     0.92     0.90      285
      1       0.77     0.67     0.71      188

accuracy                           0.85      393
macro avg       0.82     0.79     0.81      393
weighted avg    0.85     0.85     0.85      393
```

MODELO 3: RANDOM FOREST

```
Precisión: 0.8677
precision    recall   f1-score  support
      0       0.89     0.93     0.91      285
      1       0.80     0.69     0.74      188

accuracy                           0.87      393
macro avg       0.84     0.81     0.83      393
weighted avg    0.86     0.87     0.86      393
```

COMPARACIÓN FINAL:

```
=====
1. Random Forest: 0.8677
2. Árbol de Decisión: 0.8524
3. Regresión Logística: 0.8448

El mejor modelo fue: Random Forest
```

Ejercicio11:

```
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio11$ python3 Ejercicio11.py
== EJERCICIO 11: ANÁLISIS DE SENTIMIENTO - ROTTEN TOMATOES ==

Dataset cargado: 10662 reseñas
Distribución de etiquetas:
labels
0    5331
1    5331
Name: count, dtype: int64

Vectorizando texto con TF-IDF...

MODELO 1: NAIVE BAYES
-----
Precisión: 0.7546
      precision    recall   f1-score   support
Negativo       0.76      0.75      0.76     1615
Positivo       0.75      0.76      0.75     1584

accuracy          0.75      0.75      0.75     3199
macro avg       0.75      0.75      0.75     3199
weighted avg    0.75      0.75      0.75     3199

MODELO 2: REGRESIÓN LOGÍSTICA
-----
Precisión: 0.7415
      precision    recall   f1-score   support
Negativo       0.75      0.74      0.74     1615
Positivo       0.74      0.74      0.74     1584

accuracy          0.74      0.74      0.74     3199
macro avg       0.74      0.74      0.74     3199
weighted avg    0.74      0.74      0.74     3199

MODELO 3: RANDOM FOREST
-----
Precisión: 0.6968
      precision    recall   f1-score   support
Negativo       0.68      0.76      0.72     1615
Positivo       0.72      0.64      0.68     1584

accuracy          0.70      0.70      0.70     3199
macro avg       0.70      0.70      0.70     3199
weighted avg    0.70      0.70      0.70     3199

COMPARACIÓN FINAL:
=====
1. Naive Bayes: 0.7546
2. Regresión Logística: 0.7415
3. Random Forest: 0.6968

El mejor modelo fue: Naive Bayes
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio11$ |
```

```
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio12$ python3 Ejercicio12.py
== EJERCICIO 12: DETECCIÓN DE ANOMALÍAS - SALARIOS ==

Dataset cargado: 35 empleados

Primeras filas:
   ID Experience_Years  Age  Gender  Salary
0   1              5    28 Female  250000
1   2              1    21  Male   50000
2   3              3    23 Female  170000
3   4              2    22  Male   25000
4   5              1    17  Male   10000

Estadísticas descriptivas:
   ID  Experience_Years      Age      Salary
count  35.000000  35.00000  3.500000e+01
mean   18.000000  9.20000  35.485714  2.059147e+06
std    10.246951  7.55295  14.643552  3.170124e+06
min    1.000000  1.00000  17.000000  3.000000e+03
25%   9.500000  2.50000  22.500000  2.250000e+04
50%  18.000000  6.00000  29.000000  2.500000e+05
75%  26.500000  15.00000 53.500000  3.270000e+06
max   35.000000  27.00000 62.000000  1.000000e+07

Usando características: ['Experience_Years', 'Age', 'Salary']

Entrenando Isolation Forest...
Empleados normales: 33
Empleados anómalos: 2

Generando visualización...
Gráfico guardado como 'anomalias_salarios.png'

EMPLEADOS CON SALARIOS ANÓMALOS:
=====
   Experience_Years  Age  Gender  Salary
5            25    62  Male  5001000
27           27    62 Female 10000000

Rango de salarios anómalos: $5,001,000 - $10,000,000
Salario promedio general: $2,059,147
Salario promedio anómalos: $7,500,500

EJERCICIO 12 COMPLETADO
(.venv) tunek@tunek:~/Proyectos/Universidad/InteligenciaArtificial/aprendiendo_py/EjerciciosPropuestos/Ejercicio12$ |
```

