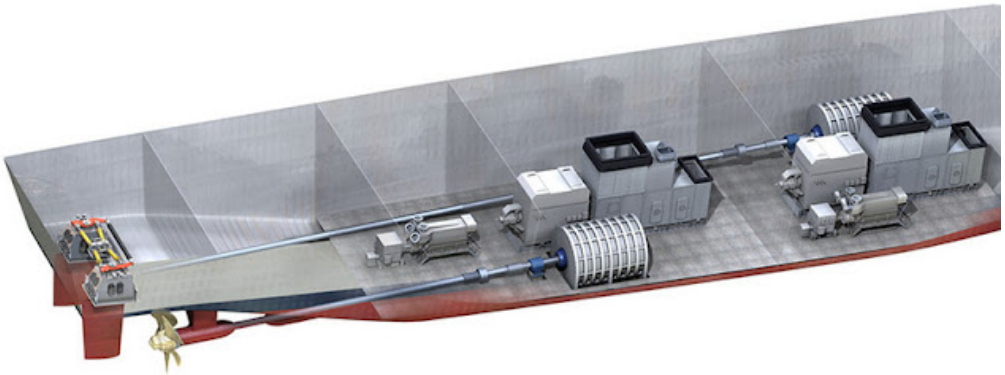
Facultad de Ingeniería
Doctorado en Ingeniería de Sistemas Complejos

Trabajo n° 1:
Algebra Lineal y Optimización para DS

Dataset:
Condition Based Maintenance of Naval Propulsion Plants



Alumno: Eduardo Carrasco
Profesor: PhD Martín Ríos-Wilson.

2023

I. Presentación de la Base de Datos:

La base de datos fue obtenida desde el Repositorio de la UCI[1] (University of California Irvine), y detalla una simulación de los parámetros de operación de una Turbina a Gas SM1C (16 parámetros y 11934 registros), de una fragata que utiliza una planta propulsora del tipo CODLAG (*Combined Diesel Electric and Gas*), es decir, utiliza una propulsión mixta entre 2 motores eléctricos (2980 KW cada uno) y 2 turbinas a gas (19500 KW).

Lo importante del set de datos es que se puede establecer una relación de degradación del rendimiento (*decay*), tanto en el compresor (*'Compressor degradation coefficient kMc'*), como en las turbina (*'Turbine degradation coefficient kMt'*), considerando el siguiente detalle de variables:

Variables Independientes	Variables Dependientes
Lever position (lp): Posición de la palanca. Ship speed (knots) GT shaft torque (GTT): [kN m]. GT rate of revolutions (GTn): [rpm]. Gas Generator rate of revolutions (GGn): [rpm]. Starboard Propeller Torque (Ts):[kN]. Port Propeller Torque (Tp): [kN]. HP Turbine exit temperature (T48): [°C]. GT Compressor inlet air temperature (T1): [°C]. GT Compressor outlet air temperature (T2): [°C]. HP Turbine exit pressure (P48): [bar]. GT Compressor inlet air pressure (P1): [bar]. GT Compressor outlet air pressure (P2): [bar]. GT exhaust gas pressure (Pexh): [bar]. Turbine Injection Control (TIC): [%]. Fuel flow (mf): [kg/s].	**GT Compressor decay state coefficient**: Coeficiente de estado de degradación del compresor de la GT. **GT Turbine decay state coefficient**: Coeficiente de estado de degradación de la turbina de la GT.

Tabla 1: Detalle de Variables Independientes y Dependientes del conjunto de datos utilizado.

Para lo anterior y con el objeto de reconocer las variables que son linealmente dependientes unas de otras y seleccionar sólo las relevantes para la resolución

[1] Coraddu, Andrea; Oneto, Luca; Ghio, Alessandro; Savio, Stefano; Anguita, Davide; and Figari, Massimo. (2014). Condition Based Maintenance of Naval Propulsion Plants. UCI Machine Learning Repository. https://doi.org/10.24432/C5K31K.

del problema, se realizó una matriz de correlación (coeficiente de pearson), de acuerdo al siguiente detalle:

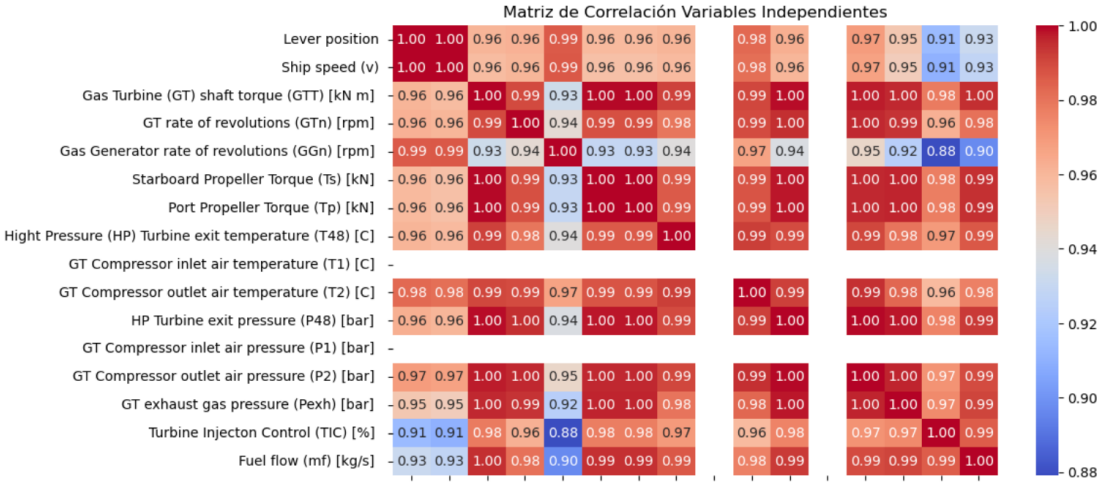


Figura 1: Matriz de correlación que representa las 16 variables independientes del dataset (Fuente: Confección propia).

Como se observa en la matriz, existen variables que tienen dependencia lineal e incluso algunas variables son constantes como en este caso *"GT Compressor inlet air (temperature / pressure)*, debido a que son las temperaturas y presiones normales de funcionamiento de una GT (*Gas Turbine*) bajo el ciclo de Brayton.
También se pueden eliminar la mayoría que son dependientes unas de otras, por ejemplo las relacionadas a revoluciones que asocian el eje central de torque propulsor con las hélices (*propellers*), tanto del eje de babor como estribor y, por otra parte, se pueden hacer relaciones directas entre presión y temperatura (mayor presión, mayor temperatura), eliminando un total de 9 variables. Con lo anterior, se obtiene la siguiente nueva matriz de correlación:

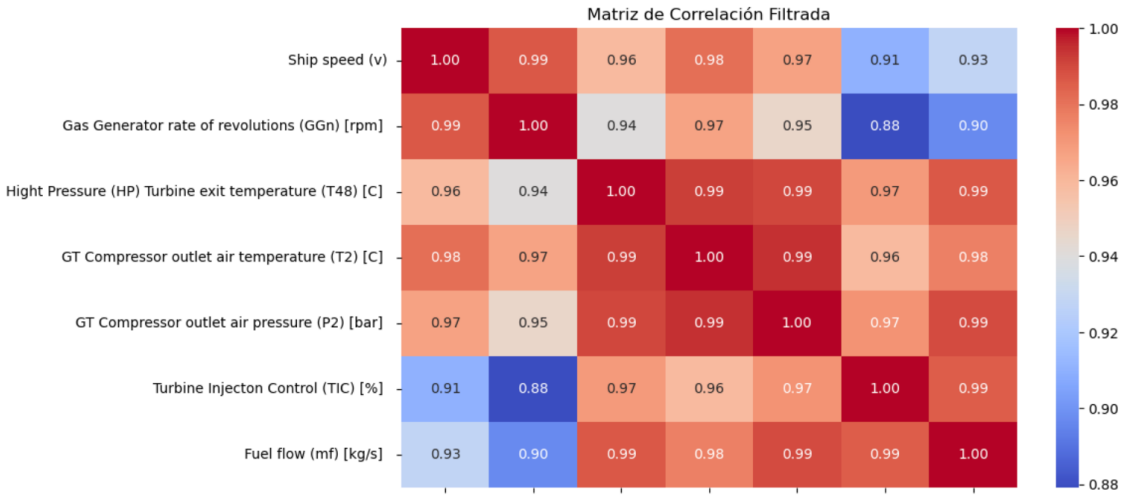


Figura 2: Matriz de correlación filtrada que representa las 7 variables independientes del conjunto de datos (Fuente: Confección propia).

II. Desarrollo de las preguntas:

A. Investigue sobre el método gráfico “Caras de Chernoff” y represente gráficamente un conjunto de variables mediante las caras de Chernoff. ¿Qué representa cada cara?, ¿Qué representan los rasgos de cara (identifique)?

Considerando que la cantidad de muestras es de 11.934 y las caras de Chernoff grafican cada una de las filas, se hace impracticable graficar la totalidad. Por lo anterior, se efectuó una discretización de datos creando una nueva variable (*GT Turbine decay state coefficient_class*) que contiene 3 clases (*Normal*, *Preventive* y *Urgent*), a partir de los siguientes bins: 0 hasta 0.981 (sin incluir 0.981), 0.981 hasta 0.994 (sin incluir 0.994) y 0.994 hasta 1 (incluyendo 1); obteniendo las siguientes cantidades:

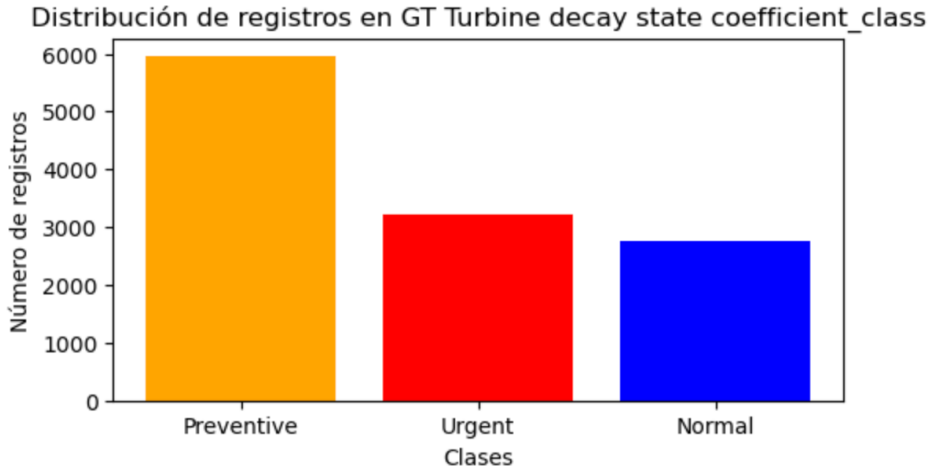


Figura 3: Gráfico de distribución de etiquetas de clasificación en DataSet - Versión 1 (Fuente: Confección propia).

Esto con el objetivo de verificar si al obtener 3 muestras diferentes de estas clases, se podían diferenciar mediante las caras de Chernoff, obteniendo el siguiente resultado:

Figura 4: Gráficos de “Caras de Chernoff” de una muestra del Dataset (Fuente: Confección propia).

Del análisis de las caras, se puede concluir que en definitiva entre todos hay una diferencia marcada pero también se observan ciertas similitudes. Entre los que parecieran ser los más extremos (*Normal* y *Urgent*), existen ciertas similaridades relacionadas a la posición de los ojos, cejas y forma de cabeza, pero se pueden distinguir colores marcados que indican una diferencia significativa en las clases.
La principal prestación que se observa de esta técnica es la relacionada a identificar patrones o anomalías de manera más intuitiva que con otras técnicas.
Cabe destacar que las clases anteriores fueron asignadas de manera arbitraria, solo utilizando los cuartiles para la definición de bins.

B. Técnicas de Reducción de Dimensionalidad:

1) Investigue y defina el método "t-SNE" como técnica de reducción de dimensionalidad.

El t-SNE (t-Distributed Stochastic Neighbor Embedding) es una técnica de Unsupervised Learning que es ampliamente utilizada para reducir la dimensionalidad de un conjunto de datos de alta dimensión. El objetivo, al igual que todas las técnicas similares, es definir una representación en menor dimensión que preserve las relaciones de vecindad entre los puntos en el espacio original.
Para lo anterior, la técnica se encarga de disminuir la divergencia entre 2 distribuciones: midiendo puntos similares en el espacio de alta dimensión (distancia de unos a otros) y lo mismo en el espacio de baja dimensión; preservando así la estructura de los datos (a nivel global y local).

2) Describa sus ventajas y desventajas respecto del método de componentes principales.

Ventajas	Desventajas
Preserva las relaciones de vecindad, lo que significa que los puntos cercanos en el espacio original también lo serán en el espacio reducido.	Los resultados pueden variar entre diferentes ejecuciones del algoritmo debido a la inicialización aleatoria
Puede capturar relaciones no lineales en los datos.	Es más sencillo (matemática y computacionalmente), por lo cual, es más adecuado para grandes conjuntos de datos.

Tabla 2: Cuadro de ventajas y desventajas entre la técnica PCA y la técnica t-SNE (Reducción de Dimensionalidad).

3) **Aplique un análisis de componentes principales y el método "t-SNE" y compare los resultados obtenidos. Sea detallado al momento de describir los hallazgos obtenidos.**

Considerando que anteriormente tomamos una de las variables dependientes y efectuamos una discretización generando 3 nuevas etiquetas en base a los cuartiles. Lo conveniente de usar PCA y t-SNE sería comparar cuántas agrupaciones pueden distinguir y cómo se diferencian estas de las definidas arbitrariamente por el autor de este informe.

Previo a los gráficos es conveniente señalar que es difícil distinguir grupos en la operación de una turbina puesto que están muy relacionados entre sí, considerando que más demanda de velocidad, conlleva a una más presión de salida, más temperatura y mayor consumo de combustible. La operación normal se da generalmente en una velocidad media que no es tan exigente para la turbina siendo las bajas velocidades y las altas velocidades las que producen estados anómalos y los gráficos nos permitirían distinguir la cantidad de estados totales que se podrían obtener.

Efectuando un PCA y un t-SNE, se obtienen los siguientes gráficos:

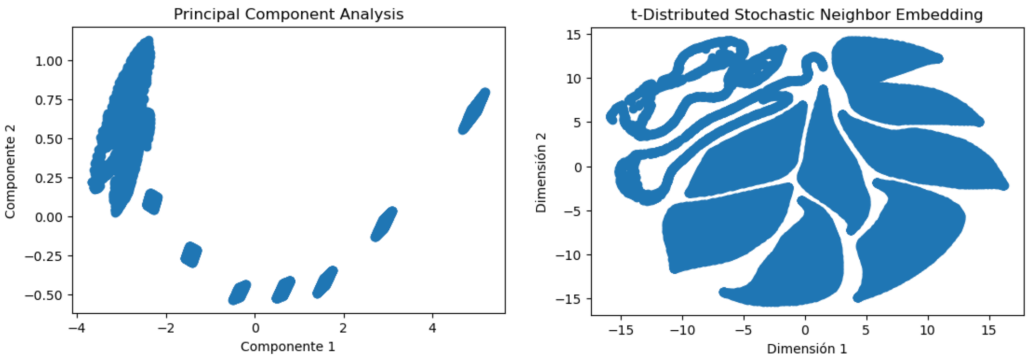


Figura 5: Gráficos de PCA y t-SNE del set de datos completo
(Fuente: Confección propia a partir de los datos).

Del análisis de los gráficos se puede observar que el PCA muestra una agrupación bastante distinguible y que el resto de los estados tienen una menor concentración de registros pero aún así, distinguibles entre sí (por la distancia mostrada entre los grupos).

Por otro lado, el t-SNE permite distinguir clases pero la distancia entre estas agrupaciones no representa que tan distintas son.

En ambos, si comparamos la cantidad de agrupaciones que cada uno pudo determinar, se puede concluir que existen entre 8 o 9 agrupaciones de datos a diferencia de las 3 definidas arbitrariamente por el autor de este documento considerando como referencia los cuartiles.

C. Técnicas de Clustering:

1) Investigue y defina sobre "Gaussian Mixture" para clasificación de observaciones.

Un Gaussian Mixture Model (GMM) es un modelo probabilístico que asume que los datos provienen de una mezcla de varias distribuciones gaussianas. Se utiliza comúnmente en tareas de clasificación para asignar observaciones a clases específicas. En este contexto, cada distribución gaussiana representa una clase diferente, y el modelo estima la probabilidad de que una observación dada pertenezca a cada clase. A diferencia de otros métodos de clasificación, GMM proporciona una asignación de clase más flexible y probabilística, lo que permite una mejor gestión de la incertidumbre y la ambigüedad en los datos.

2) Establezca una regla de discriminante lineal para su base de datos seleccionada. Indique si tiene información previa de las probabilidades de pertenencia a los determinados grupos (probabilidades a priori), de ser así, uselas en su regla de clasificación. Grafique sus resultados (ambas reglas de clasificación).

Considerando que la regla de discriminante lineal es un sistema de clasificación binario, se efectuó un nuevo discretizado de manera de generar 2 grandes grupos (*Operational "1"* y Critical "0"), a partir de los siguientes bins: 0 hasta 0.9875 (sin incluir 0.9875) y 0.9875 hasta 1 (incluyendo 1); obteniendo las siguientes cantidades:

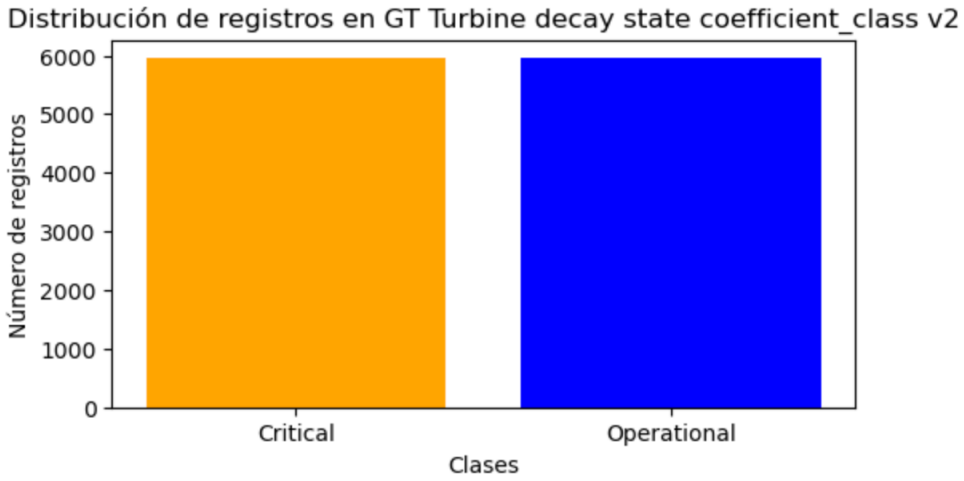


Figura 6: Gráfico de distribución de etiquetas de clasificación en DataSet - Versión 2 (Fuente: Confección propia).

Esta clasificación binomial (discriminante lineal) es arbitraria y sólo está basada en la media de los datos, al igual que el ejemplo anterior, no se cuenta con información de las etiquetas de clasificación ni de cuántos clústers existen. Sólo se asegura de mantener la proporción de uno y otro lado de las etiquetas (5967 *Operational* "1" y 5967 *Critical "0"*). Luego y con el propósito de efectuar un LDA, GMM y la validación, se decidió dividir el conjunto de datos en un 80 y 20, tal como se detalla posteriormente en punto 4).

3) **Use "Gaussian Mixture" en la base de datos seleccionada. Compare sus resultados con los obtenidos en el ítem anterior. Sea detallado al momento de describir los hallazgos obtenidos.**
Una vez dividido el conjunto de datos y realizado el LDA, se utilizó el GMM, definiendo de manera sencilla mediante el ajuste del modelo a dos distribuciones gaussianas de manera de obtener etiquetas. Al implementar el modelo, se obtuvieron 5279 etiquetas *Critical "0"* y 4268 *Operational "1"*. Pero al efectuar métricas, se pudo observar lo siguiente:

```
Informe de clasificación:
              precision    recall  f1-score   support

           0       0.49      0.60      0.54      4315
           1       0.59      0.48      0.53      5232

    accuracy                           0.54      9547
   macro avg       0.54      0.54      0.54      9547
weighted avg       0.55      0.54      0.54      9547
```

Figura 7: Cuadro resumen de métricas del modelo de clasificación (Fuente: Confección propia).

El informe indica que el modelo tiene un rendimiento equilibrado pero mediocre en la clasificación de las clases *Critical "0"* y *Operational "1"*. Con una precisión y exactitud del 49% y 59% correspondientemente, el modelo es poco efectivo. Esto probablemente se debe a que no sólo existen 2 clases como fue definido arbitrariamente por el autor del documento.

4) **Simule 10 nuevas observaciones. De acuerdo a la regla de clasificación mediante discriminante lineal y Gaussian Mixture, clasifique estas 10 nuevas observaciones. Grafique sus resultados y compárelos.**

Con el objeto de completar este paso y tal como fue detallado en el punto a), se decidió dividir el set de datos principal en 80 y 20, inicialmente se utilizaron 9547 registros para el desarrollo de la pregunta 2) y 3); para luego tomar una muestra del segundo conjunto de datos (2387), de 10 registros y validar el modelo.

Como se puede observar en el siguiente informe de clasificación, el rendimiento general del modelo es regular, esto comparando lo obtenido por el discriminante lineal y el GMM.

Informe de clasificación:

	precision	recall	f1-score	support
0	0.60	0.75	0.67	4
1	0.80	0.67	0.73	6
accuracy			0.70	10
macro avg	0.70	0.71	0.70	10
weighted avg	0.72	0.70	0.70	10

Figura 8: Cuadro resumen de métricas del modelo de clasificación con nuevos registros (Fuente: Confección propia).

III. Conclusiones:

Este estudió abordó un problema técnico asociado a la operación de una turbina a gas, dentro de una planta de propulsión naval, con el uso de datos simulados similares al sensorizado actualmente implementado en estas máquinas de combustión. Inicialmente se efectuó un preprocesamiento, de manera de determinar las variables más relevantes para el problema y separar el conjunto de variables dependientes e independientes.

La incorporación de las caras de Chernoff ofreció una forma visualmente intuitiva de entender la variabilidad y las diferencias entre los registros de datos, acompañado para lo anterior, de una discretización de manera de generar unas pseudo-etiquetas, que permitieran interpretar fácilmente las diferencias de un conjunto de datos.

Posteriormente, se implementaron técnicas de reducción de dimensionalidad basadas tanto en PCA como en t-SNE, permitiendo comprender la estructura interna de los datos y mostrando gráficamente los clusters que se agrupan de manera diferente a nuestra decisión anterior. PCA por una parte permitió observar una visión lineal y de distancia entre clusters y t-SNE otorgando una perspectiva no lineal.

Finalmente, se utilizaron técnicas de clustering, tanto Discriminante Lineal como GMM. Inicialmente para la implementación del clustering, como no se tiene información de los grupos existentes en el conjunto de datos se definieron en forma arbitraria 2 grupos: Operational “1” y Critical “0”, para luego generar nuevas etiquetas utilizando estas 2 técnicas. El resultado final demuestra una deficiente capacidad para hacer predicciones, probablemente asociado a que no existen sólo 2 clústers, por lo cual, su definición es fundamental para el resultado del modelo.

IV. Referencias bibliográficas:

A. Abbas, O. A. (2008). Comparisons between data clustering algorithms. International Arab Journal of Information Technology (IAJIT), 5(3).

B. Anowar, F., Sadaoui, S., & Selim, B. (2021). Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne). Computer Science Review, 40, 100378.

C. Charman, N., & Rand, M. J. (2003). Development and In Service Introduction of a Rolls-Royce Spey SM1C Digital Control System. In Turbo Expo: Power for Land, Sea, and Air (Vol. 3686, pp. 447-453).

D. Cipollini, F., Oneto, L., Coraddu, A., Murphy, A. J., & Anguita, D. (2018). Condition-based maintenance of naval propulsion systems with supervised data analysis. Ocean Engineering, 149, 268-278.

E. Coraddu, A., Oneto, L., Ghio, A., Savio, S., Anguita, D., & Figari, M. (2016). Machine learning approaches for improving condition-based maintenance of naval propulsion plants. Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment, 230(1), 136-153.

F. Prajapati, A., Bechtel, J., & Ganesan, S. (2012). Condition based maintenance: a survey. Journal of Quality in Maintenance Engineering, 18(4), 384-400.

G. Lee, M. D., Reilly, R. E., & Butavicius, M. E. (2003, January). An empirical evaluation of Chernoff faces, star glyphs, and spatial visualizations for binary data. In Proceedings of the Asia-Pacific symposium on Information visualization-Volume 24 (pp. 1-10).

H. Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari & Lin, C. T. (2017). A review of clustering techniques and developments. Neurocomputing, 267, 664-681.

V. Código Utilizado:

Se encuentra publicado en el siguiente repositorio:
https://github.com/educarrascov/DISC_Algebra/blob/main/Trabajo%201.ipynb

Dataset: Condition Based Maintenance of Naval Propulsion Plants

Eduardo Carrasco Estudiante Doctorado en Ingeniería de Sistemas Complejos

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import pearsonr

df = pd.read_csv('Dataset/data.csv', sep=',')
df
```

Out[1]:

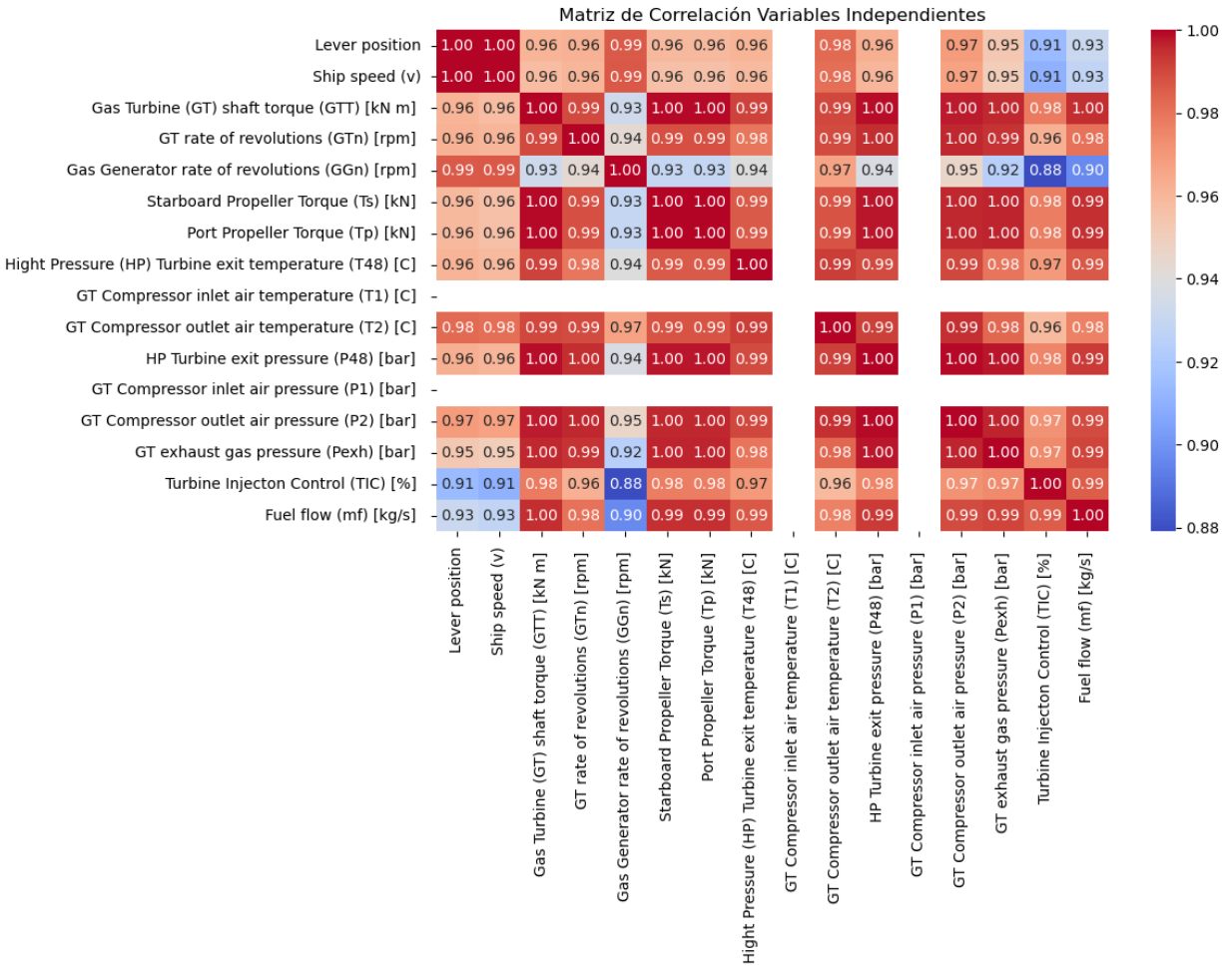
	Lever position	Ship speed (v)	Gas Turbine (GT) shaft torque (GTT) [kN m]	GT rate of revolutions (GTn) [rpm]	Gas Generator rate of revolutions (GGn) [rpm]	Starboard Propeller Torque (Ts) [kN]	Port Propeller Torque (Tp) [kN]	Hight Pressure (HP) Turbine exit temperature (T48) [C]
0	1.138	3	289.964	1349.489	6677.380	7.584	7.584	464.006
1	2.088	6	6960.180	1376.166	6828.469	28.204	28.204	635.401
2	3.144	9	8379.229	1386.757	7111.811	60.358	60.358	606.002
3	4.161	12	14724.395	1547.465	7792.630	113.774	113.774	661.471
4	5.140	15	21636.432	1924.313	8494.777	175.306	175.306	731.494
...	...	...	...	...	...	...	...	...
11929	5.140	15	21624.934	1924.342	8470.013	175.239	175.239	681.658
11930	6.175	18	29763.213	2306.745	8800.352	245.954	245.954	747.405
11931	7.148	21	39003.867	2678.052	9120.889	332.389	332.389	796.457
11932	8.206	24	50992.579	3087.434	9300.274	438.024	438.024	892.945
11933	9.300	27	72775.130	3560.400	9742.950	644.880	644.880	1038.411

11934 rows × 18 columns

In [2]:
```
df.columns
```

Out[2]:
```
Index(['Lever position ', 'Ship speed (v) ',
       'Gas Turbine (GT) shaft torque (GTT) [kN m]  ',
       'GT rate of revolutions (GTn) [rpm]  ',
       'Gas Generator rate of revolutions (GGn) [rpm]  ',
       'Starboard Propeller Torque (Ts) [kN]  ',
       'Port Propeller Torque (Tp) [kN]  ',
       'Hight Pressure (HP) Turbine exit temperature (T48) [C]  ',
       'GT Compressor inlet air temperature (T1) [C]  ',
       'GT Compressor outlet air temperature (T2) [C]  ',
       'HP Turbine exit pressure (P48) [bar]  ',
       'GT Compressor inlet air pressure (P1) [bar]  ',
       'GT Compressor outlet air pressure (P2) [bar]  ',
       'GT exhaust gas pressure (Pexh) [bar]  ',
       'Turbine Injecton Control (TIC) [%]  ', 'Fuel flow (mf) [kg/s]  ',
       'GT Compressor decay state coefficient  ',
       'GT Turbine decay state coefficient '],
      dtype='object')
```

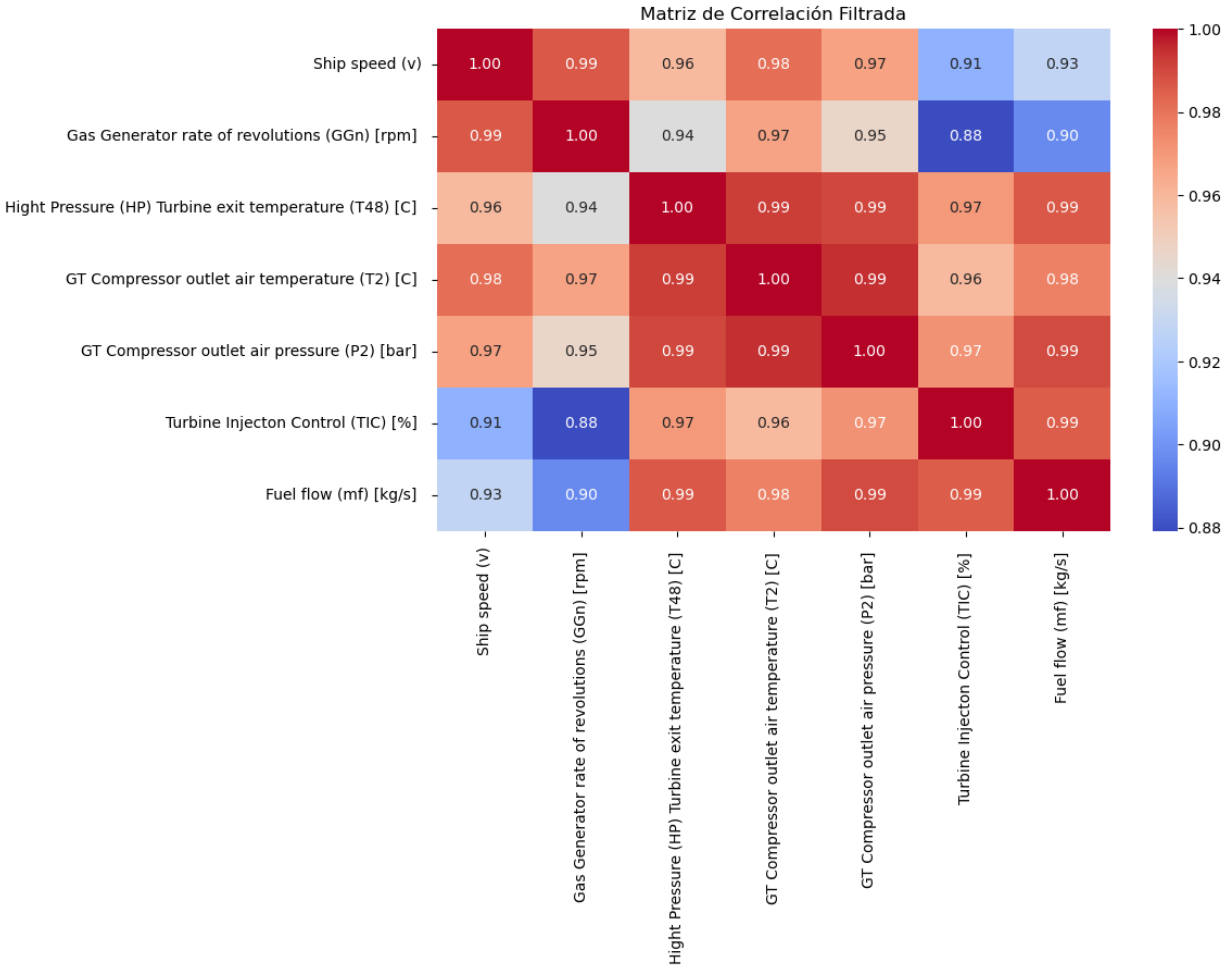
In [3]:
```
df_independientes = df.iloc[:, :16]
matriz_correlacion = df_independientes.corr()

plt.figure(figsize=(10, 6))
sns.heatmap(matriz_correlacion, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Matriz de Correlación Variables Independientes')
plt.show()
```



In [4]:
```
df_filtrado = df_independientes.drop(df_independientes.columns[[0, 2, 3, 5, 6,
print(df_filtrado.columns)
```

```
Index(['Ship speed (v) ', 'Gas Generator rate of revolutions (GGn) [rpm]  ',
       'Hight Pressure (HP) Turbine exit temperature (T48) [C]  ',
       'GT Compressor outlet air temperature (T2) [C]  ',
       'GT Compressor outlet air pressure (P2) [bar]  ',
       'Turbine Injecton Control (TIC) [%]  ', 'Fuel flow (mf) [kg/s]  '],
      dtype='object')
```

In [5]:
```
matriz_correlacion = df_filtrado.corr()

plt.figure(figsize=(10, 6))
sns.heatmap(matriz_correlacion, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Matriz de Correlación Filtrada')
plt.show()
```



In [6]:
```
df1=df
print(df1[df1.columns[17]].describe())
```

```
count    11934.0000
mean         0.9875
std          0.0075
min          0.9750
25%          0.9810
50%          0.9875
75%          0.9940
max          1.0000
Name: GT Turbine decay state coefficient , dtype: float64
```

In [7]:
```
bins = [0, 0.981, 0.994, 1]
labels = ['Urgent', 'Preventive', 'Normal']
```

```
df1['GT Turbine decay state coefficient_class'] = pd.cut(df1['GT Turbine decay
                                                    bins=bins, labels=label
```

In [8]:
```
df1.columns
```

Out[8]:
```
Index(['Lever position ', 'Ship speed (v) ',
       'Gas Turbine (GT) shaft torque (GTT) [kN m]  ',
       'GT rate of revolutions (GTn) [rpm]  ',
       'Gas Generator rate of revolutions (GGn) [rpm]  ',
       'Starboard Propeller Torque (Ts) [kN]  ',
       'Port Propeller Torque (Tp) [kN]  ',
       'Hight Pressure (HP) Turbine exit temperature (T48) [C]  ',
       'GT Compressor inlet air temperature (T1) [C]  ',
       'GT Compressor outlet air temperature (T2) [C]  ',
       'HP Turbine exit pressure (P48) [bar]  ',
       'GT Compressor inlet air pressure (P1) [bar]  ',
       'GT Compressor outlet air pressure (P2) [bar]  ',
       'GT exhaust gas pressure (Pexh) [bar]  ',
       'Turbine Injecton Control (TIC) [%]  ', 'Fuel flow (mf) [kg/s]  ',
       'GT Compressor decay state coefficient  ',
       'GT Turbine decay state coefficient ',
       'GT Turbine decay state coefficient_class'],
      dtype='object')
```

In [9]:
```
df1 = df1.drop(df.columns[[0, 2, 3, 5, 6, 8, 10, 11, 13, 16]], axis=1)
df1
```

Out[9]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT Turbine decay state coefficient
0	3	6677.380	464.006	550.563	5.947	7.137	0.082	0.975
1	6	6828.469	635.401	581.658	7.282	10.655	0.287	0.975
2	9	7111.811	606.002	587.587	7.574	13.086	0.259	0.975
3	12	7792.630	661.471	613.851	9.007	18.109	0.358	0.975
4	15	8494.777	731.494	645.642	11.197	26.373	0.522	0.975
...	...	...	...	...	...	...	...	...
11929	15	8470.013	681.658	628.950	10.990	23.803	0.471	1.000
11930	18	8800.352	747.405	658.853	13.109	32.671	0.647	1.000
11931	21	9120.889	796.457	680.393	15.420	42.104	0.834	1.000
11932	24	9300.274	892.945	722.029	18.293	58.064	1.149	1.000
11933	27	9742.950	1038.411	767.595	22.464	86.067	1.704	1.000

11934 rows × 9 columns

In [10]:
```
df1.describe(include='all')
```

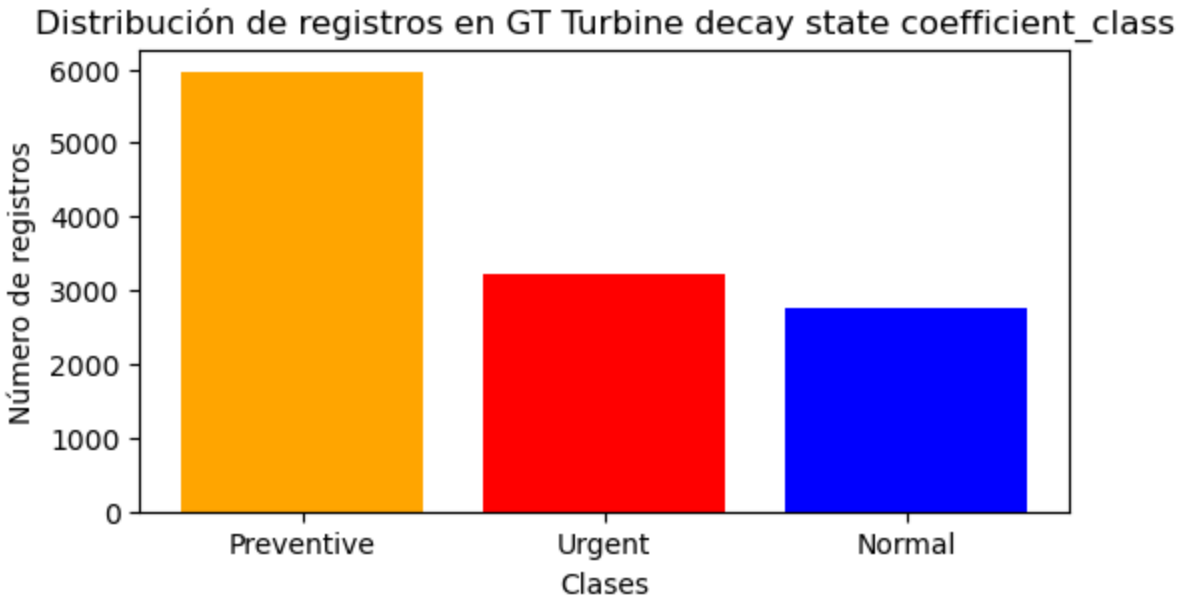
Out[10]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]
count	11934.000000	11934.000000	11934.000000	11934.000000	11934.000000	11934.000000
unique	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN	NaN
mean	15.000000	8200.947312	735.495446	646.215331	12.297123	33.641261
std	7.746291	1091.315507	173.680552	72.675882	5.337448	25.841363
min	3.000000	6589.002000	442.364000	540.442000	5.828000	0.000000
25%	9.000000	7058.324000	589.872750	578.092250	7.447250	13.677500
50%	15.000000	8482.081500	706.038000	637.141500	11.092000	25.276500
75%	21.000000	9132.606000	834.066250	693.924500	15.658000	44.552500
max	27.000000	9797.103000	1115.797000	789.094000	23.140000	92.556000

In [11]:
```
df1.columns
```

Out[11]:
```
Index(['Ship speed (v) ', 'Gas Generator rate of revolutions (GGn) [rpm]  ',
       'Hight Pressure (HP) Turbine exit temperature (T48) [C]  ',
       'GT Compressor outlet air temperature (T2) [C]  ',
       'GT Compressor outlet air pressure (P2) [bar]  ',
       'Turbine Injecton Control (TIC) [%]  ', 'Fuel flow (mf) [kg/s]  ',
       'GT Turbine decay state coefficient ',
       'GT Turbine decay state coefficient_class'],
      dtype='object')
```

In [13]:
```
import matplotlib.pyplot as plt

column_name = "GT Turbine decay state coefficient_class"

class_count = df1[column_name].value_counts()

plt.figure(figsize=(6, 3))
plt.bar(class_count.index, class_count.values, color=['orange', 'red', 'blue'])

plt.xlabel('Clases')
plt.ylabel('Número de registros')
plt.title(f'Distribución de registros en {column_name}')

plt.show()
```



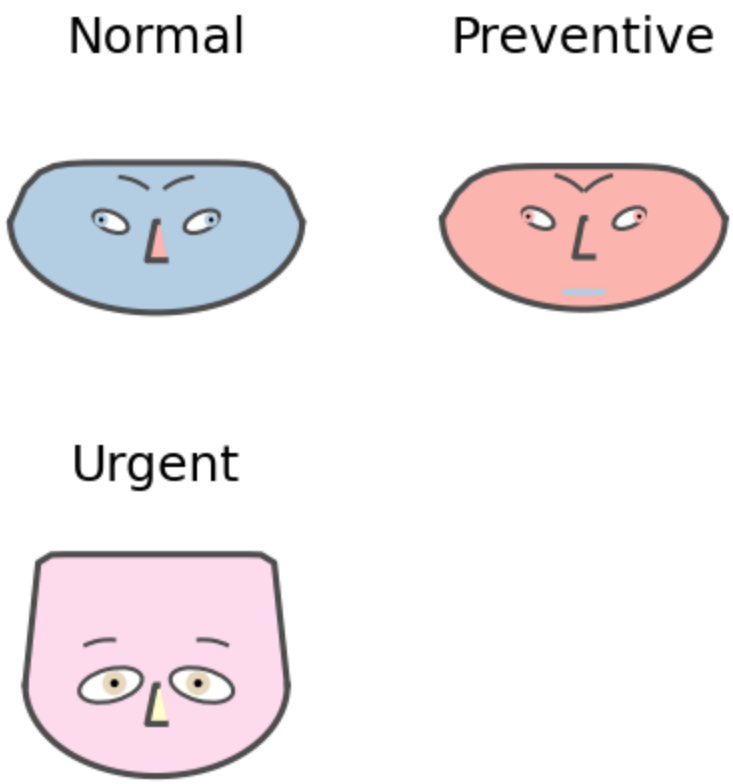
In [14]:
```
count_labels = df1['GT Turbine decay state coefficient_class'].value_counts()

print(count_labels)
```

```
Preventive    5967
Urgent        3213
Normal        2754
Name: GT Turbine decay state coefficient_class, dtype: int64
```

Chernoff Faces:

In [16]:
```
import numpy as np
import matplotlib.pyplot as plt
from ChernoffFace import chernoff_face
import matplotlib
from sklearn.preprocessing import MinMaxScaler
import pandas as pd

df_to_normalize = df1.iloc[:, :7]
scaler = MinMaxScaler()
df_normalized = pd.DataFrame(scaler.fit_transform(df_to_normalize), columns=df_
df_normalized['GT Turbine decay state coefficient_class'] = df1['GT Turbine dec

sample_urgent = df_normalized[df_normalized['GT Turbine decay state coefficient
sample_preventive = df_normalized[df_normalized['GT Turbine decay state coeffic
sample_normal = df_normalized[df_normalized['GT Turbine decay state coefficient

sample_df = pd.concat([sample_urgent, sample_preventive, sample_normal])

data = sample_df.iloc[:, :7].to_numpy()

titles = sample_df['GT Turbine decay state coefficient_class'].to_list()

fig = chernoff_face(data=data,
                    titles=titles,
                    color_mapper=plt.cm.Pastel1,
                    figsize=(3, 3), dpi=150)
```

```
fig.tight_layout()
plt.show()
```



In [17]:
```
sample_urgent
```

Out[17]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT Tu decay coefficient
416	0.25	0.1784	0.218057	0.176705	0.092479	0.133897	0.10034	N

In [18]:
```
sample_preventive
```

Out[18]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT dec coefficie
8299	0.125	0.06349	0.172717	0.106084	0.064464	0.0	0.073696	P

In [19]:
```
sample_normal
```

Out[19]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT dec coefficier
970	0.875	0.85041	0.748259	0.802447	0.736657	0.669767	0.657029	

In [20]:
```
sample_df
```

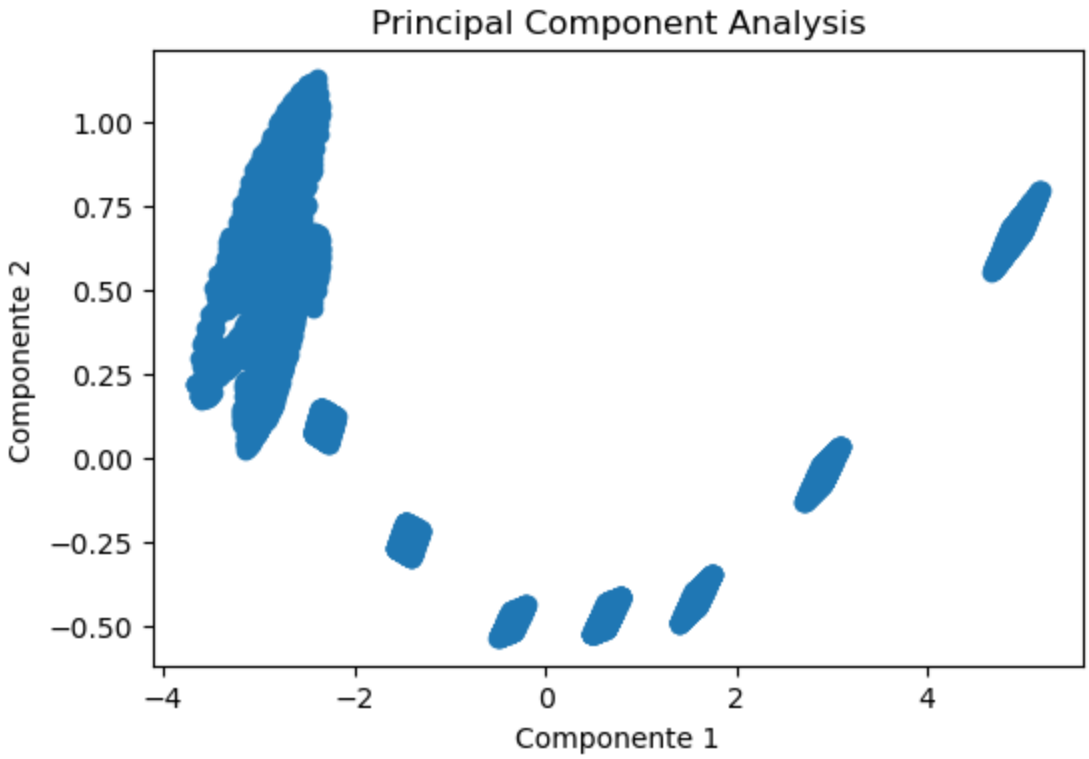
Out[20]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	G de coeffici
416	0.250	0.17840	0.218057	0.176705	0.092479	0.133897	0.100340	
8299	0.125	0.06349	0.172717	0.106084	0.064464	0.000000	0.073696	
970	0.875	0.85041	0.748259	0.802447	0.736657	0.669767	0.657029	

Análisis de Componentes Principales:

PCA:

In [21]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df_normalized = scaler.fit_transform(df_filtrado)

pca = PCA(n_components=2)
pca_result = pca.fit_transform(df_normalized)

plt.figure(figsize=(6, 4))
plt.scatter(pca_result[:, 0], pca_result[:, 1])
plt.title('Principal Component Analysis')
plt.xlabel('Componente 1')
plt.ylabel('Componente 2')
plt.show()
```

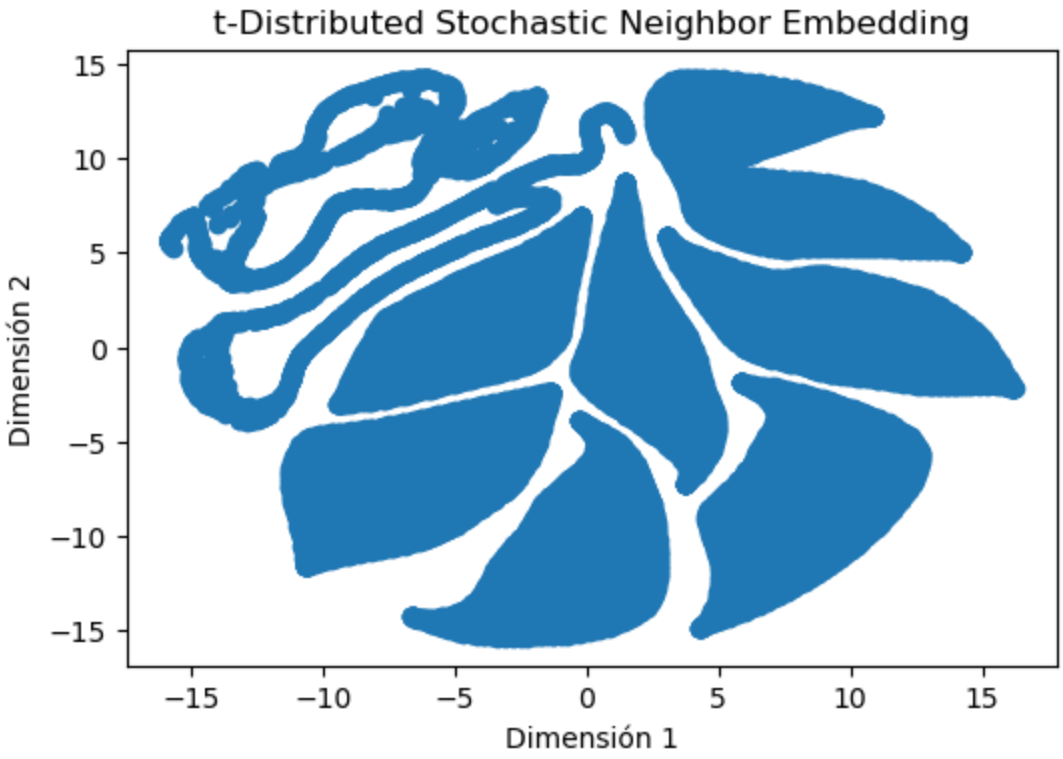


In [22]:
```
pca_result
```

Out[22]:
```
array([[-3.47190838,  0.35886895],
       [-2.43445719,  0.44790835],
       [-2.19103928,  0.12216411],
       ...,
       [ 1.39162151, -0.49384466],
       [ 2.70154371, -0.13241672],
       [ 4.67498613,  0.55419661]])
```

t-SNE:

In [23]:
```
tsne = TSNE(n_components=2, perplexity=30, n_iter=300)
tsne_result = tsne.fit_transform(df_normalized)

plt.figure(figsize=(6, 4))
plt.scatter(tsne_result[:, 0], tsne_result[:, 1])
plt.title('t-Distributed Stochastic Neighbor Embedding')
plt.xlabel('Dimensión 1')
plt.ylabel('Dimensión 2')
plt.show()
```



In [24]:
```
tsne_result
```

Out[24]:
```
array([[-12.679729 ,   8.562398 ],
       [  1.4311563,  11.301516 ],
       [ -0.1987351,   6.955657 ],
       ...,
       [  4.2806506, -14.847783 ],
       [ 16.201    ,  -2.173212 ],
       [ 14.189847 ,   5.054433 ]], dtype=float32)
```

Técnicas de Clustering:

Clasificación Binomial:

In [41]:
```
df2=df
df2
```

Out[41]:

	Lever position	Ship speed (v)	Gas Turbine (GT) shaft torque (GTT) [kN m]	GT rate of revolutions (GTn) [rpm]	Gas Generator rate of revolutions (GGn) [rpm]	Starboard Propeller Torque (Ts) [kN]	Port Propeller Torque (Tp) [kN]	Hight Pressure (HP) Turbine exit temperature (T48) [C]
0	1.138	3	289.964	1349.489	6677.380	7.584	7.584	464.006
1	2.088	6	6960.180	1376.166	6828.469	28.204	28.204	635.401
2	3.144	9	8379.229	1386.757	7111.811	60.358	60.358	606.002
3	4.161	12	14724.395	1547.465	7792.630	113.774	113.774	661.471
4	5.140	15	21636.432	1924.313	8494.777	175.306	175.306	731.494
...	...	...	...	...	...	...	...	...
11929	5.140	15	21624.934	1924.342	8470.013	175.239	175.239	681.658
11930	6.175	18	29763.213	2306.745	8800.352	245.954	245.954	747.405
11931	7.148	21	39003.867	2678.052	9120.889	332.389	332.389	796.457
11932	8.206	24	50992.579	3087.434	9300.274	438.024	438.024	892.945
11933	9.300	27	72775.130	3560.400	9742.950	644.880	644.880	1038.411

11934 rows × 19 columns

In [42]:
```
print(df2[df2.columns[17]].describe())
```

```
count    11934.0000
mean         0.9875
std          0.0075
min          0.9750
25%          0.9810
50%          0.9875
75%          0.9940
max          1.0000
Name: GT Turbine decay state coefficient , dtype: float64
```

In [43]:
```
bins = [0, 0.987530, 1]
labels = [0, 1]
df2['GT Turbine decay state coefficient_class'] = pd.cut(df2['GT Turbine decay
                                                    bins=bins, labels=label
```

In [44]:
```
df2 = df2.drop(df.columns[[0, 2, 3, 5, 6, 8, 10, 11, 13, 16]], axis=1)
df2
```

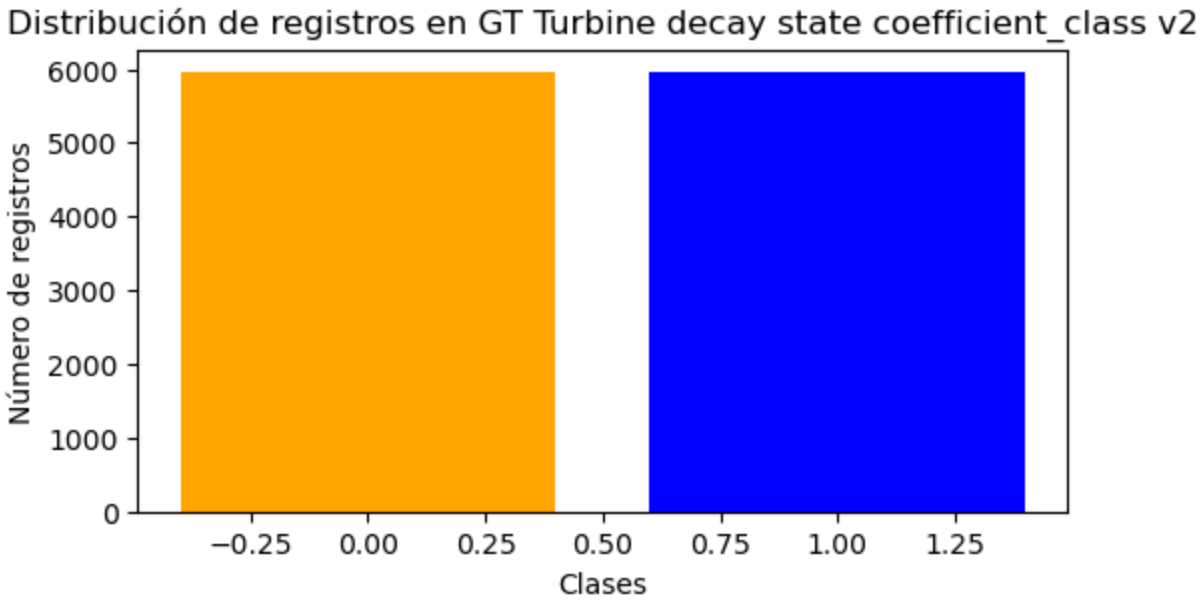
Out[44]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT Turbine decay state coefficient
0	3	6677.380	464.006	550.563	5.947	7.137	0.082	0.975
1	6	6828.469	635.401	581.658	7.282	10.655	0.287	0.975
2	9	7111.811	606.002	587.587	7.574	13.086	0.259	0.975
3	12	7792.630	661.471	613.851	9.007	18.109	0.358	0.975
4	15	8494.777	731.494	645.642	11.197	26.373	0.522	0.975
...	...	...	...	...	...	...	...	...
11929	15	8470.013	681.658	628.950	10.990	23.803	0.471	1.000
11930	18	8800.352	747.405	658.853	13.109	32.671	0.647	1.000
11931	21	9120.889	796.457	680.393	15.420	42.104	0.834	1.000
11932	24	9300.274	892.945	722.029	18.293	58.064	1.149	1.000
11933	27	9742.950	1038.411	767.595	22.464	86.067	1.704	1.000

11934 rows × 9 columns

In [45]:

```
import matplotlib.pyplot as plt

column_name = "GT Turbine decay state coefficient_class"

class_count = df2[column_name].value_counts()

plt.figure(figsize=(6, 3))
plt.bar(class_count.index, class_count.values, color=['orange', 'blue'])

plt.xlabel('Clases')
plt.ylabel('Número de registros')
plt.title(f'Distribución de registros en {column_name} v2')

plt.show()
```



In [46]:
```
count_labels = df2['GT Turbine decay state coefficient_class'].value_counts()

print(count_labels)
```

```
0    5967
1    5967
Name: GT Turbine decay state coefficient_class, dtype: int64
```

División 2 Set datos:

In [47]:
```
from sklearn.model_selection import train_test_split

df2_1, df2_2 = train_test_split(df2, test_size=0.2, random_state=42)
```

Uso Set de Datos 2_1

In [62]:
```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.mixture import GaussianMixture
import pandas as pd

X = df2_1.iloc[:, :7]

y = df2_1['GT Turbine decay state coefficient_class']

lda = LinearDiscriminantAnalysis(n_components=1)
lda.fit(X, y)
lda_labels = lda.predict(X)

df2_1['LDA_Labels'] = lda_labels

gmm = GaussianMixture(n_components=2)
gmm.fit(X)
gmm_labels = gmm.predict(X)

df2_1['GMM_Labels'] = gmm_labels
```

In [63]:
```
df2_1
```

Out[63]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT Turbine decay state coefficient
3685	15	8497.497	704.972	638.429	11.004	24.951	0.494	0.994
5886	3	6622.351	545.800	557.982	6.450	26.222	0.194	0.979
6108	21	9120.644	835.034	691.451	15.730	44.789	0.887	0.977
7713	3	6697.636	620.925	570.351	7.078	0.000	0.278	1.000
3499	24	9319.460	917.854	733.407	18.180	59.577	1.179	0.999
...	...	...	...	...	...	...	...	...
11284	24	9292.800	915.018	725.973	18.693	60.083	1.189	0.980
5191	24	9306.729	933.160	734.056	18.621	61.190	1.211	0.979
5390	27	9751.200	1089.292	780.775	22.961	90.760	1.797	0.975
860	18	8842.649	779.989	671.876	13.129	34.453	0.682	0.992
7270	24	9300.348	930.374	731.780	18.715	61.133	1.210	0.976

9547 rows × 11 columns

In [64]:
```
count_labels = df2_1['GMM_Labels'].value_counts()

print(count_labels)
```

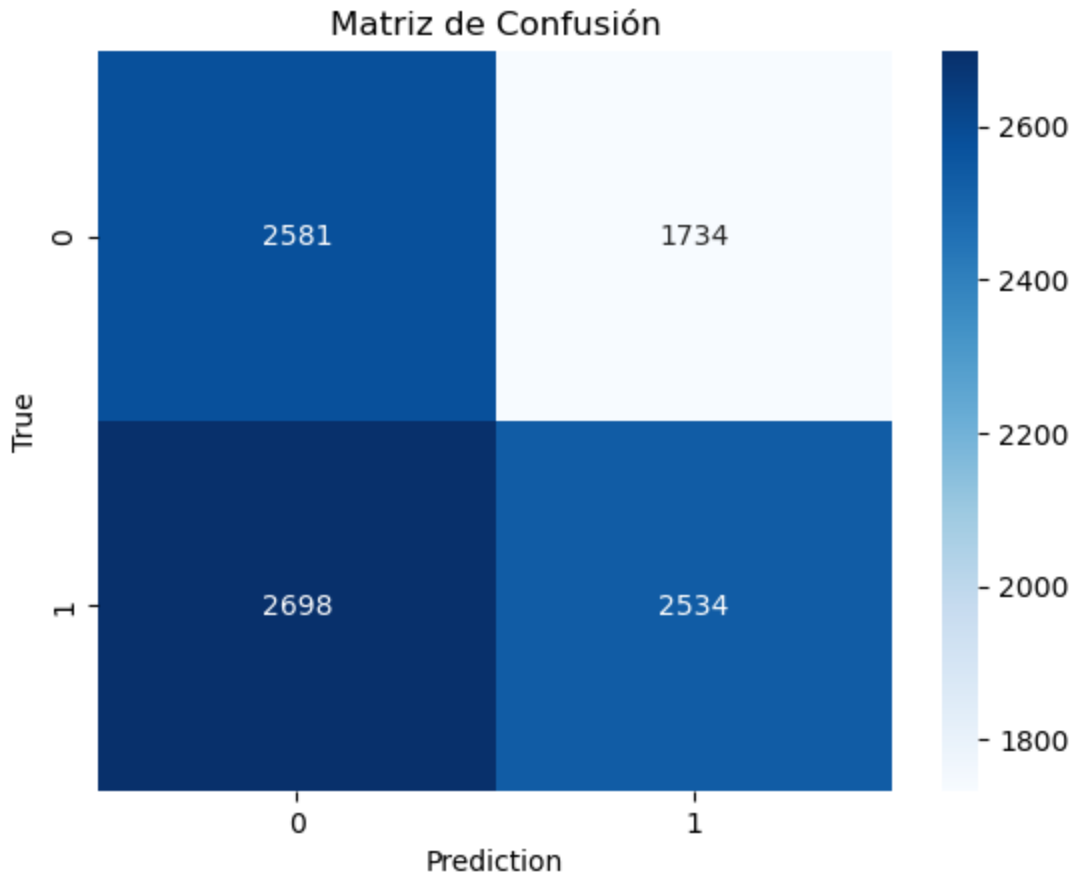
```
0    5279
1    4268
Name: GMM_Labels, dtype: int64
```

In [65]:
```
count_labels = df2_1['LDA_Labels'].value_counts()

print(count_labels)
```

```
1    5232
0    4315
Name: LDA_Labels, dtype: int64
```

In [66]:
```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

y_true = df2_1['LDA_Labels']

y_pred = df2_1['GMM_Labels']

conf_matrix = confusion_matrix(y_true, y_pred, labels=[0, 1])

sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Prediction')
plt.ylabel('True')
```

```
plt.title('Matriz de Confusión')
plt.show()
```



In [67]:
```
from sklearn.metrics import classification_report

report = classification_report(y_true, y_pred, labels=[0, 1])

print("Informe de clasificación:")
print(report)
```

```
Informe de clasificación:
              precision    recall  f1-score   support

           0       0.49      0.60      0.54      4315
           1       0.59      0.48      0.53      5232

    accuracy                           0.54      9547
   macro avg       0.54      0.54      0.54      9547
weighted avg       0.55      0.54      0.54      9547
```

Uso del Set de datos 2_2:

In [68]:
```
df2_2
```

Out[68]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT Turbine decay state coefficient
10172	9	7045.840	576.879	576.548	7.506	11.961	0.237	0.987
10322	27	9728.709	1064.615	773.249	23.045	88.928	1.760	0.977
9195	21	9117.010	821.362	686.661	15.687	43.923	0.870	0.982
357	21	9138.462	840.619	697.393	15.460	44.791	0.887	0.988
4352	18	8824.273	774.265	668.631	13.185	34.208	0.677	0.990
...	...	...	...	...	...	...	...	...
7388	27	9754.242	1068.077	775.818	22.632	88.646	1.755	0.989
7176	12	7757.985	632.714	604.558	8.882	16.839	0.333	0.992
5721	21	9126.158	827.086	690.517	15.569	44.125	0.874	0.986
267	21	9133.424	850.929	699.107	15.630	45.600	0.903	0.978
6086	9	7052.874	594.000	581.883	7.586	12.645	0.250	0.975

2387 rows × 9 columns

In [86]:
```
sample_critical = df2_2[df2_2['GT Turbine decay state coefficient_class'] == 0]
sample_operational = df2_2[df2_2['GT Turbine decay state coefficient_class'] ==
sample_df2_2 = pd.concat([sample_critical, sample_operational])
sample_df2_2
```

Out[86]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT Turbine decay state coefficient
2874	12	7778.879	649.580	610.050	8.955	17.587	0.348	0.982
11562	21	9113.862	811.930	683.191	15.669	43.335	0.858	0.985
5170	15	8475.422	717.707	639.763	11.207	25.730	0.509	0.977
4453	24	9306.912	939.413	735.678	18.695	61.703	1.222	0.975
10537	24	9292.106	922.313	727.726	18.785	60.689	1.201	0.975
428	18	8848.979	776.761	671.511	13.071	34.224	0.678	0.996
6962	18	8816.658	764.322	665.117	13.152	33.643	0.666	0.994
5271	21	9128.132	826.380	690.793	15.529	44.034	0.872	0.988
2714	18	8831.833	777.918	670.338	13.172	34.387	0.681	0.990
668	9	7164.169	587.161	583.864	7.415	12.310	0.244	0.997

In [87]:
```
X_new = sample_df2_2.iloc[:, :7]
```

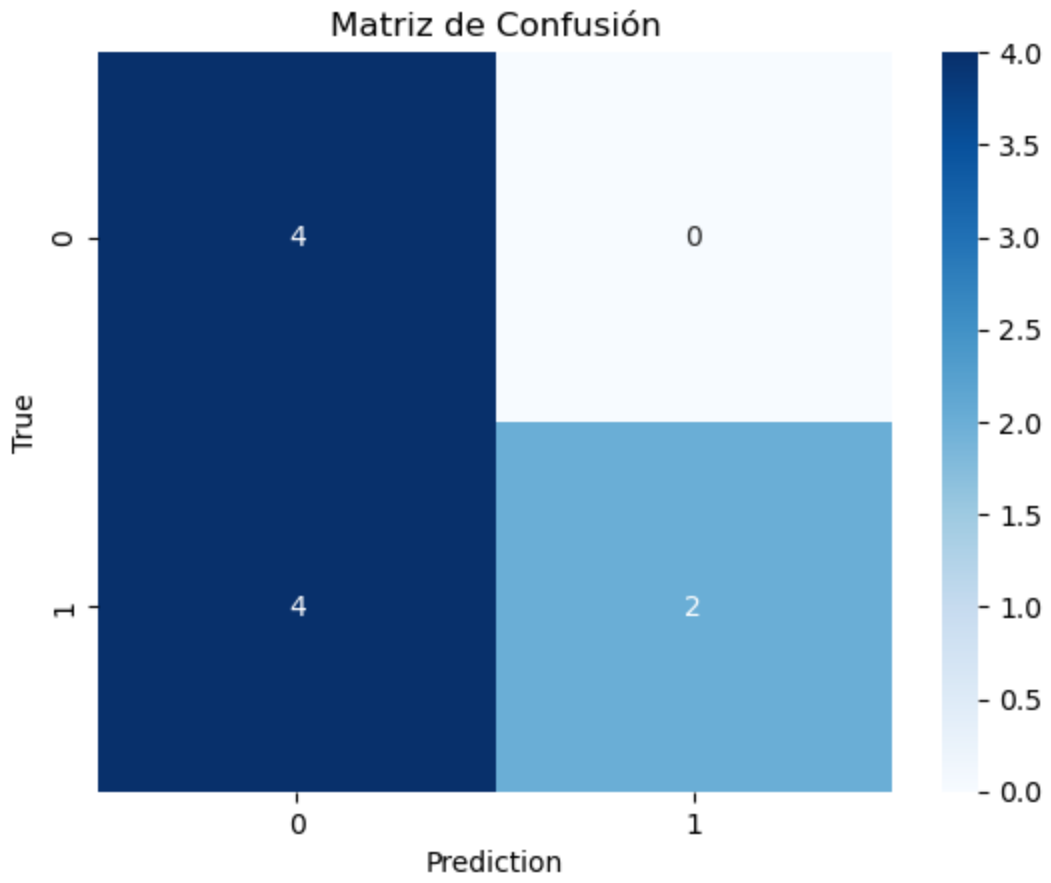
```
new_lda_labels = lda.predict(X_new)

new_gmm_labels = gmm.predict(X_new)

sample_df2_2['New_LDA_Labels'] = new_lda_labels
sample_df2_2['New_GMM_Labels'] = new_gmm_labels
```

In [88]:
```
sample_df2_2
```

Out[88]:

	Ship speed (v)	Gas Generator rate of revolutions (GGn) [rpm]	Hight Pressure (HP) Turbine exit temperature (T48) [C]	GT Compressor outlet air temperature (T2) [C]	GT Compressor outlet air pressure (P2) [bar]	Turbine Injecton Control (TIC) [%]	Fuel flow (mf) [kg/s]	GT Turbine decay state coefficient
2874	12	7778.879	649.580	610.050	8.955	17.587	0.348	0.982
11562	21	9113.862	811.930	683.191	15.669	43.335	0.858	0.985
5170	15	8475.422	717.707	639.763	11.207	25.730	0.509	0.977
4453	24	9306.912	939.413	735.678	18.695	61.703	1.222	0.975
10537	24	9292.106	922.313	727.726	18.785	60.689	1.201	0.975
428	18	8848.979	776.761	671.511	13.071	34.224	0.678	0.996
6962	18	8816.658	764.322	665.117	13.152	33.643	0.666	0.994
5271	21	9128.132	826.380	690.793	15.529	44.034	0.872	0.988
2714	18	8831.833	777.918	670.338	13.172	34.387	0.681	0.990
668	9	7164.169	587.161	583.864	7.415	12.310	0.244	0.997

In [89]:
```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

y_true = sample_df2_2['New_LDA_Labels']

y_pred = sample_df2_2['New_GMM_Labels']

conf_matrix = confusion_matrix(y_true, y_pred, labels=[0, 1])

sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Prediction')
plt.ylabel('True')
plt.title('Matriz de Confusión')
plt.show()
```



In [90]:
```
from sklearn.metrics import classification_report

report = classification_report(y_true, y_pred, labels=[0, 1])

print("Informe de clasificación:")
print(report)
```

```
Informe de clasificación:
              precision    recall  f1-score   support

           0       0.50      1.00      0.67         4
           1       1.00      0.33      0.50         6

    accuracy                           0.60        10
   macro avg       0.75      0.67      0.58        10
weighted avg       0.80      0.60      0.57        10
```

In [ ]: