PROPUESTA DE SOLUCIÓN AL ODS: SALUD Y BIENESTAR: Plataforma de Bienestar Estudiantil Basada en IA

La plataforma utilizaría el modelo de machine learning para analizar continuamente los datos de matrícula y académicos de los estudiantes en tiempo real. Cada vez que un estudiante se matricula en un nuevo ciclo, la plataforma puede evaluar automáticamente su riesgo de problemas de salud mental.

Impacto:
Esta plataforma no solo podría ayudar a identificar y apoyar a los estudiantes en riesgo, sino que también fomentar un ambiente universitario más consciente y proactivo en la promoción de la salud mental. Además, podría servir como un modelo que se pueda replicar para otras instituciones educativas interesadas en integrar soluciones de IA en sus programas de bienestar estudiantil.

¿La propuesta es aplicable y escalable?

Sí, la propuesta es aplicable y escalable. Puede adaptarse a diversas universidades, integrar nuevas funcionalidades y expandirse globalmente usando tecnología en la nube.

Contribución al ODS

La solución contribuye al ODS de Salud y Bienestar al identificar y prevenir problemas de salud mental en estudiantes, permitiendo intervenciones tempranas. Al analizar factores de riesgo, el modelo promueve un entorno académico más saludable, reduciendo el estrés y mejorando el bienestar emocional.

¿Es viable técnicamente y económicamente?

La propuesta es técnicamente viable, utilizando machine learning con recursos moderados. Económicamente, es costeable y sostenible, aprovechando datos existentes y reduciendo costos futuros en salud mental.

Beneficio a la comunidad

Mi propuesta mejorará la detección temprana de riesgos de salud mental en estudiantes, permitiendo intervenciones personalizadas. Esto contribuirá a un ambiente académico más saludable, reduciendo deserciones y mejorando el bienestar general de la comunidad estudiantil.

Modelo de Machine Learning:

1. Importación de Librerías

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, roc_auc_score, confusion_matrix
```

Descripción: Importa las librerías necesarias para el análisis de datos y la modelización.

2. Lectura y Preprocesamiento de Datos

```
df_est = pd.read_csv('C:\\Users\\CRISTIAN\\Documents\\Documents\\HACKATONS\\Dataton Expresate con datos\\Datos_abiertos_matricula
```

```
df_est.head(10)
```

	IDHASH	COLEGIO_DEPA	COLEGIO_PROV	COLEGIO_DIST	ANIO	PERIODO	TIPO_MATRICULA	DOMICILIO
0	07E35E858686718757529DDDBEC110B0B39CA2FCC810A6...	LIMA	LIMA METROPOLITANA	LURIGANCHO	2016	1	Regulares	
1	24DAB8EC1FDFA72428AF843FFABB4901743386A9EBD0A1...	LIMA	LIMA METROPOLITANA	ATE	2016	1	Regulares	
2	0FD5293AAD1655B7FBF28920796B1C49144F4B7F3BEA24...	LIMA	LIMA METROPOLITANA	VILLA EL SALVADOR	2016	1	Regulares	
3	5B07CA8222FAB9610D2B3C0D3789CAF1103479F47ACA8E...	ÁNCASH	CARHUAZ	CARHUAZ	2016	1	Regulares	Á
4	FF657E45CD5AE985DCDF4E3C7B5CCA17F90056F075473C...	LIMA	LIMA METROPOLITANA	CARABAYLLO	2016	1	Regulares	AMA
5	5588CFE4A9A61A76B0462F2EC1B1A28D271B273CA0B855...	LIMA	LIMA METROPOLITANA	VILLA MARIA DEL TRIUNFO	2016	1	Regulares	
6	822BF1B04078B8D6AA41B0387D547D3EDB078C6817C4D7...	LIMA	LIMA METROPOLITANA	ATE	2016	1	Regulares	
7	288D0B9B092FABA10B3F868FC9FD1FC22A40960B2A3626...	LIMA	LIMA METROPOLITANA	COMAS	2016	1	Regulares	
8	02C7FA590F72F5372F25B56C203BA889077832FCC7FBBD	LIMA	LIMA METROPOLITANA	SAN MARTIN DE PORRES	2016	1	Regulares	AMA

```
df_est.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 192389 entries, 0 to 192388
Data columns (total 21 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   IDHASH           192389 non-null  object
 1   COLEGIO_DEPA     188686 non-null  object
 2   COLEGIO_PROV     188541 non-null  object
 3   COLEGIO_DIST     188442 non-null  object
 4   ANIO             192389 non-null  int64
 5   PERIODO          192389 non-null  int64
 6   TIPO_MATRICULA   192389 non-null  object
 7   DOMICILIO_DEPA   190898 non-null  object
 8   DOMICILIO_PROV   170767 non-null  object
 9   DOMICILIO_DIST   170523 non-null  object
 10  ANIO_NACIMIENTO  192389 non-null  int64
 11  NACIMIENTO_PAIS  192202 non-null  object
 12  NACIMIENTO_DEPA  191749 non-null  object
 13  NACIMIENTO_PROV  191672 non-null  object
 14  NACIMIENTO_DIST  191672 non-null  object
 15  SEXO             192389 non-null  object
 16  MODALIDAD        192389 non-null  object
 17  METODOLOGIA      192389 non-null  object
 18  FACULTAD         192389 non-null  object
 19  ESPECIALIDAD     192389 non-null  object
 20  CICLO_RELATIVO   192389 non-null  int64
dtypes: int64(4), object(17)
memory usage: 30.8+ MB
```

In [6]:
```
# Ver el porcentaje de valores nulos en cada columna
porcentaje_nulos = df_est.isnull().sum() / len(df_est) * 100
porcentaje_nulos
```

Out[6]:
```
IDHASH               0.000000
COLEGIO_DEPA         1.924746
COLEGIO_PROV         2.000114
COLEGIO_DIST         2.051573
ANIO                 0.000000
PERIODO              0.000000
TIPO_MATRICULA       0.000000
DOMICILIO_DEPA       0.774992
DOMICILIO_PROV      11.238688
DOMICILIO_DIST      11.365515
ANIO_NACIMIENTO      0.000000
NACIMIENTO_PAIS      0.097199
NACIMIENTO_DEPA      0.332659
NACIMIENTO_PROV      0.372682
NACIMIENTO_DIST      0.372682
SEXO                 0.000000
MODALIDAD            0.000000
METODOLOGIA          0.000000
FACULTAD             0.000000
ESPECIALIDAD         0.000000
CICLO_RELATIVO       0.000000
dtype: float64
```

In [7]:
```
# Eliminar valores nulos de aquellas columnas menores al 5 % que no tomaré en cuenta en el modelo predictivo
columnas_a_eliminar = porcentaje_nulos[porcentaje_nulos < 5].index
df_est = df_est.dropna(subset=columnas_a_eliminar)

# Elimino las columnas DOMICILIO_PROV Y DOMICILIO_DIST  ya que no necesito dichas columnas para mi análisis
df_est = df_est.drop(columns = ['DOMICILIO_PROV', 'DOMICILIO_DIST'])
```

In [8]:
```
df_est[df_est.isnull().any(axis=1)]
```

Descripción: Carga el archivo CSV, revisa la información y el porcentaje de valores nulos, y elimina columnas y filas con valores nulos según los criterios establecidos.

3. Formato Estándar de Datos

In [9]:
```
# Estableciendo formato estandar de las modalidades de ingreso
df_est.loc[(df_est['MODALIDAD'] == 'CONCURSO NACIONAL ESCOLAR'), 'MODALIDAD'] = 'INGRESO ESCOLAR NACIONAL'
df_est.loc[(df_est['MODALIDAD'] == 'CONCURSO NACIONAL ESCOLAR - PROVINCIAS'), 'MODALIDAD'] = 'CONCURSO NACIONAL ESCOLAR (OR+LP)'
df_est.loc[(df_est['MODALIDAD'] == 'CONCURSO NACIONAL ESCOLAR - LIMA Y CALLAO'), 'MODALIDAD'] = 'CONCURSO NACIONAL ESCOLAR (LM+C)
df_est.loc[
    (df_est['MODALIDAD'] == 'INGRESO DIRECTO') |
    (df_est['MODALIDAD'] == 'INGRESO DIRECTO CEPRE') |
    (df_est['MODALIDAD'] == 'INGRESO DIRECTO CEPRE-UNI O CEPRE-UNI INTENSIVO') |
    (df_est['MODALIDAD'] == 'INGRESO DIRECTO CEPRE-UNI INTENSIVO'),
    'MODALIDAD'
] = 'INGRESO DIRECTO CEPRE-UNI'
df_est.loc[df_est['MODALIDAD'] == 'DIPLOMADOS CON BACHILLERATO', 'MODALIDAD'] = 'DIPLOMADO CON BACHILLERATO INTERNACIONAL'
df_est.loc[df_est['MODALIDAD'] == 'TITULADOS O GRADUADOS EN LA UNI', 'MODALIDAD'] = 'TITULADOS O GRADUADOS UNI'
df_est.loc[
    (df_est['MODALIDAD'] == 'VICTIMA DEL TERRORISMO') |
    (df_est['MODALIDAD'] == 'VÍCTIMAS DEL TERRORISMO'),
     'MODALIDAD'
] = 'VICTIMAS DEL TERRORISMO'
```

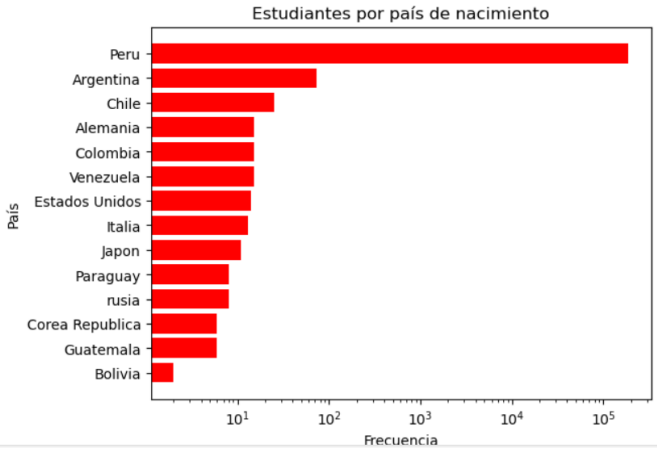
Descripción: Estandariza los valores en la columna 'MODALIDAD' para una consistencia en el análisis.

4. Análisis Exploratorio de Datos (EDA)

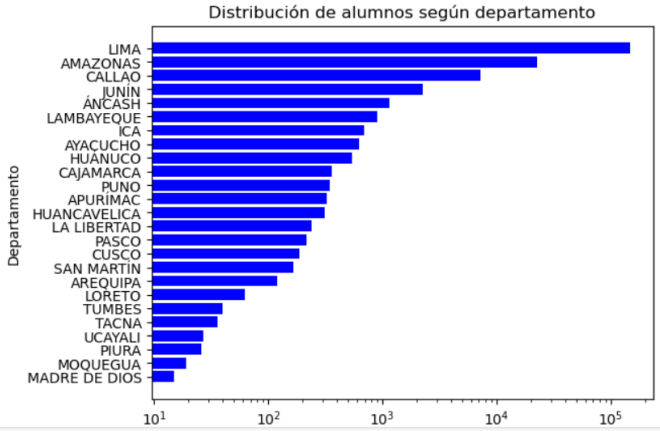
```python
# Estudiantes por facultad y especialidad
#AÑO 2021 POST PANDEMIA
df_est[df_est.ANIO == 2021].groupby(['FACULTAD', 'ESPECIALIDAD'])['IDHASH'].count().sort_values(ascending = False)
```

```
FACULTAD                                          ESPECIALIDAD
INGENIERÍA CIVIL                                  INGENIERÍA CIVIL                                2819
ARQUITECTURA, URBANISMO Y ARTES                   ARQUITECTURA                                    1467
INGENIERÍA INDUSTRIAL Y DE SISTEMAS               INGENIERÍA DE SISTEMAS                          1420
                                                  INGENIERÍA INDUSTRIAL                           1329
INGENIERÍA QUÍMICA Y TEXTIL                       INGENIERÍA QUÍMICA                              1252
INGENIERÍA ECONÓMICA, ESTADÍSTICA Y CIENCIAS SOCIALES  INGENIERÍA ECONÓMICA                       1105
INGENIERÍA ELÉCTRICA Y ELECTRÓNICA                INGENIERÍA ELECTRÓNICA                           970
                                                  INGENIERÍA ELÉCTRICA                             950
INGENIERÍA MECÁNICA                               INGENIERÍA MECATRÓNICA                           923
INGENIERÍA ELÉCTRICA Y ELECTRÓNICA                INGENIERÍA DE TELECOMUNICACIONES                 914
INGENIERÍA MECÁNICA                               INGENIERÍA MECÁNICA                              913
                                                  INGENIERÍA MECÁNICA Y ELÉCTRICA                  910
INGENIERÍA GEOLÓGICA, MINERA Y METALÚRGICA        INGENIERÍA DE MINAS                              709
CIENCIAS                                          CIENCIA DE LA COMPUTACIÓN                        643
INGENIERÍA AMBIENTAL                              INGENIERÍA SANITARIA                             601
                                                  INGENIERÍA AMBIENTAL                             575
INGENIERÍA GEOLÓGICA, MINERA Y METALÚRGICA        INGENIERÍA GEOLÓGICA                             565
INGENIERÍA ECONÓMICA, ESTADÍSTICA Y CIENCIAS SOCIALES  INGENIERÍA ESTADÍSTICA                      494
INGENIERÍA GEOLÓGICA, MINERA Y METALÚRGICA        INGENIERÍA METALÚRGICA                           473
INGENIERÍA AMBIENTAL                              INGENIERÍA DE HIGIENE Y SEGURIDAD INDUSTRIAL     463
CIENCIAS                                          FÍSICA                                           457
                                                  INGENIERÍA FÍSICA                                382
INGENIERÍA MECÁNICA                               INGENIERÍA NAVAL                                 329
CIENCIAS                                          MATEMÁTICA                                       322
INGENIERÍA DE PETRÓLEO, GAS NATURAL Y PETROQUÍMICA  INGENIERÍA PETROQUÍMICA                        259
CIENCIAS                                          QUÍMICA                                          242
INGENIERÍA QUÍMICA Y TEXTIL                       INGENIERÍA TEXTIL                                236
INGENIERÍA DE PETRÓLEO, GAS NATURAL Y PETROQUÍMICA  INGENIERÍA DE PETRÓLEO Y GAS NATURAL           231
                                                  INGENIERÍA DE PETRÓLEO                             1
Name: IDHASH, dtype: int64
```

```python
# Estudiantes por Pais de nacimiento

est_por_pais = df_est['NACIMIENTO_PAIS'].value_counts().sort_values(ascending=True)

plt.barh(est_por_pais.index, est_por_pais.values, color = 'red')
plt.xscale('log')
plt.xlabel('Frecuencia')
plt.ylabel('País')
plt.title('Estudiantes por país de nacimiento')
plt.show()
```



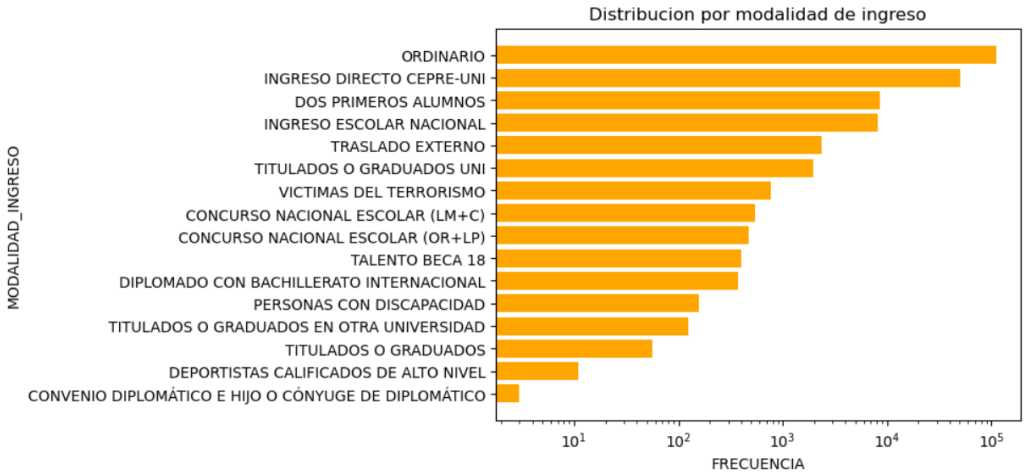
In [14]:
```
# Estudiantes por Region de procedencia
est_por_region = df_est['DOMICILIO_DEPA'].value_counts().sort_values(ascending=True)

plt.barh(est_por_region.index, est_por_region.values, color='blue')
plt.xlabel('Frecuencia')
plt.ylabel('Departamento')
plt.title('Distribución de alumnos según departamento')

plt.xscale('log')

plt.show()
```



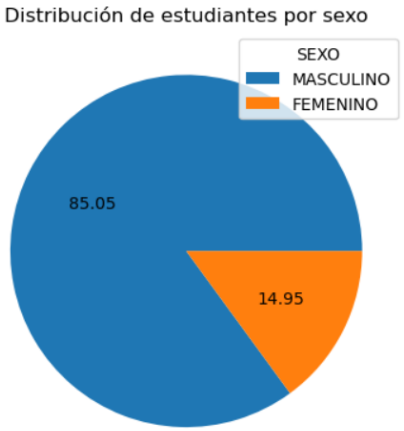

```
# Estudiantes por tipo de matricula
est_por_tipo_matricula = df_est.TIPO_MATRICULA.value_counts()
est_por_tipo_matricula
```

```
TIPO_MATRICULA
Regulares                        146908
Promoción                         19700
Cachimbos                         14710
Reincorporados Normales            3653
Riesgo Academico                   1273
Promocion                           809
Reincorporados Trikas                66
Reincorporados Riesgo Academico       8
Alumno Trika                          1
Name: count, dtype: int64
```

In [17]:
```
# Distribución de estudiantes segun modalidad de ingreso
est_por_mod_ing = df_est['MODALIDAD'].value_counts().sort_values(ascending=True)
plt.barh(est_por_mod_ing.index, est_por_mod_ing.values, color = 'orange')
plt.xscale('log')
plt.xlabel('FRECUENCIA')
plt.ylabel('MODALIDAD_INGRESO')
plt.title('Distribucion por modalidad de ingreso')
plt.show()
```



In [18]:
```
# Distribución de estudiantes por sexo
est_por_sexo = df_est['SEXO'].value_counts()
plt.pie(est_por_sexo, autopct='%.2f')
plt.legend(est_por_sexo.index, title = 'SEXO')
plt.title('Distribución de estudiantes por sexo')
plt.show()
```



Descripción: Realiza un análisis exploratorio sobre la distribución de estudiantes por facultad, especialidad, país de nacimiento, región, modalidad de ingreso y sexo.

5. Creación de Variable Objetivo

In [21]:
```
# Se creará una columna que mida la salud mental
# Se excluirá a los estudiantes del ciclo 11 por tener condicion de egresados
df_est = df_est.loc[df_est['CICLO_RELATIVO'] != 11]
# Crear la columna de problemas de salud mental basada en TIPO_MATRICULA
alto_riesgo = ['Riesgo Académico', 'Reincorporados Trikas', 'Reincorporados Riesgo Académico']
df_est['problemas_salud_mental'] = df_est['TIPO_MATRICULA'].apply(
    lambda x: 1 if x in alto_riesgo else 0)

# Ciclos bajos en combinación con tipos de matrícula de riesgo
df_est.loc[(df_est['CICLO_RELATIVO'] < 3) & (df_est['TIPO_MATRICULA'].isin(alto_riesgo)), 'problemas_salud_mental'] = 1
```

Descripción: Crea una columna para problemas de salud mental basada en el tipo de matrícula y el ciclo relativo.

6. Preparación para el Modelado

```
# Se calculará la matriz de correlación de las variables: ['SEXO', 'MODALIDAD', 'FACULTAD', 'ESPECIALIDAD', 'TIPO_MATRICULA', 'C

features = ['SEXO', 'MODALIDAD', 'FACULTAD', 'ESPECIALIDAD', 'TIPO_MATRICULA', 'CICLO_RELATIVO', 'DOMICILIO_DEPA', 'problemas_sa
df_features = df_est[features]

# Convertir variables categóricas a variables dummy
df_dummies = pd.get_dummies(df_features, drop_first=True)

# Calcular la matriz de correlación
matriz_correlacion_dummies = df_dummies.corr().sort_values(by = 'problemas_salud_mental', ascending=False)

print(matriz_correlacion_dummies['problemas_salud_mental'])
#sns.heatmap(matriz_correlacion_dummies, annot=True)
#plt.show()
```

```
problemas_salud_mental                          1.000000
TIPO_MATRICULA_Reincorporados Trikas            1.000000
MODALIDAD_TITULADOS O GRADUADOS UNI              0.092931
ESPECIALIDAD_INGENIERÍA CIVIL                    0.028353
FACULTAD_INGENIERÍA CIVIL                        0.028353
                                                   ...
FACULTAD_CIENCIAS                               -0.005900
FACULTAD_INGENIERÍA INDUSTRIAL Y DE SISTEMAS    -0.006382
MODALIDAD_INGRESO DIRECTO CEPRE-UNI             -0.006577
MODALIDAD_ORDINARIO                             -0.014432
TIPO_MATRICULA_Regulares                        -0.039348
Name: problemas_salud_mental, Length: 91, dtype: float64
```

Descripción: Prepara los datos para el modelado convirtiendo variables categóricas en dummies y calculando la matriz de correlación.

7. Entrenamiento del Modelo

In [24]:
```
label_encoders = {}
for column in ['SEXO', 'MODALIDAD', 'FACULTAD', 'ESPECIALIDAD', 'TIPO_MATRICULA', 'DOMICILIO_DEPA']:
    le = LabelEncoder()
    df_est[column] = le.fit_transform(df_est[column])
    label_encoders[column] = le
```

In [25]:
```
# Seleccionar las columnas que predictoras y la variable objetivo
features = ['SEXO', 'MODALIDAD', 'FACULTAD', 'ESPECIALIDAD', 'TIPO_MATRICULA', 'CICLO_RELATIVO', 'DOMICILIO_DEPA']
X = df_est[features]
y = df_est.problemas_salud_mental

# Dividir el conjuntos de de entrenamiento y prueba (80 / 20)
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8, random_state=1)
```

In [26]:
```
# Entrenamiento del modelo
modelo = RandomForestClassifier(random_state=1)
modelo.fit(X_train, y_train)
```

Out[26]:
```
RandomForestClassifier
RandomForestClassifier(random_state=1)
```

Descripción: Codifica las variables categóricas y entrena un modelo de clasificación con Random Forest.

8. Evaluación del Modelo

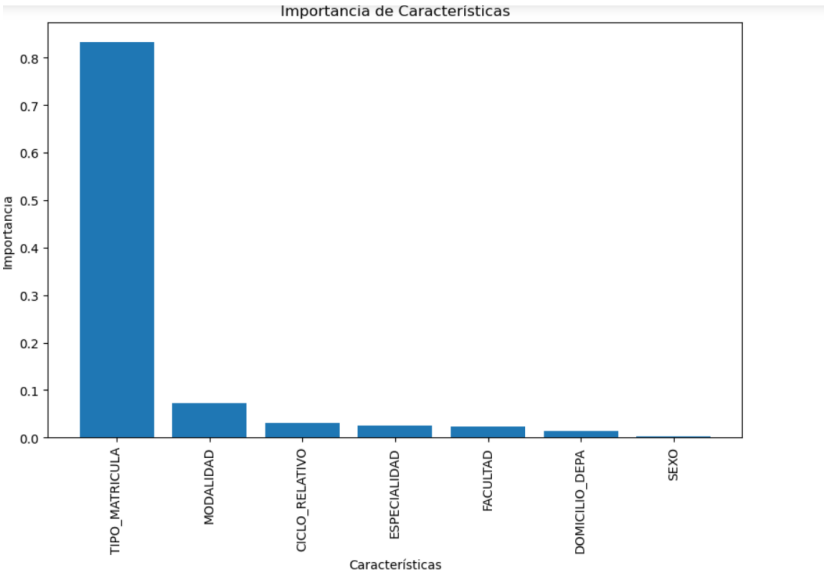
In [27]:
```
y_pred = modelo.predict(X_test)
y_prob = modelo.predict_proba(X_test)[:, 1]

# Metricas
print(classification_report(y_test, y_pred))
print(f'AUC-ROC: {roc_auc_score(y_test, y_prob)}')
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     35622
           1       1.00      1.00      1.00        13

    accuracy                           1.00     35635
   macro avg       1.00      1.00      1.00     35635
weighted avg       1.00      1.00      1.00     35635

AUC-ROC: 1.0
```

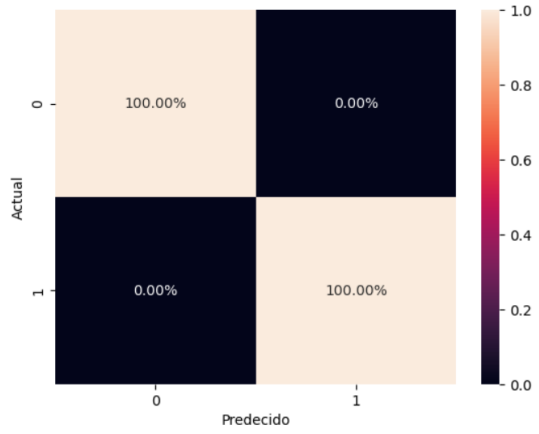
Descripción: Evalúa el rendimiento del modelo mediante métricas de clasificación y AUC-ROC.

9. Visualización de Resultados

```
# Importancia de las características en el modelo
importancias = modelo.feature_importances_
nombres_caracteristicas = X.columns
sorted_indices = np.argsort(importancias)[::-1]

plt.figure(figsize=(10, 6))
plt.title('Importancia de Características')
plt.bar(range(X.shape[1]), importancias[sorted_indices], align='center')
plt.xticks(range(X.shape[1]), nombres_caracteristicas[sorted_indices], rotation=90)
plt.xlabel('Características')
plt.ylabel('Importancia')
plt.show()
```




```
# Matriz de confusión
matriz_confusion = confusion_matrix(y_test, y_pred)

matriz_confusion_porcentajes = matriz_confusion / matriz_confusion.sum(axis = 1, keepdims=True)
sns.heatmap(matriz_confusion_porcentajes, fmt='.2%', annot=True)
plt.xlabel('Predecido')
plt.ylabel('Actual')
plt.show()
```



Descripción: Visualiza la importancia de las características y la matriz de confusión del modelo.

Hallazgos Más Importantes

- **Distribución de Estudiantes:**

  Por Facultad y Especialidad: Se observan diferencias notables en la cantidad de estudiantes por facultad y especialidad entre los años 2021, 2022 y 2023, lo que refleja cambios en la demanda de ciertas especialidades.

  Por País de Nacimiento: La mayoría de los estudiantes provienen de países con una alta frecuencia en el dataset, lo que se muestra claramente en el gráfico de barras.

- **Distribución por Modalidad de Ingreso:**

  Las modalidades de ingreso más comunes están dominadas por opciones como 'INGRESO DIRECTO CEPRE-UNI' y 'CONCURSO NACIONAL ESCOLAR', mientras que las modalidades menos frecuentes incluyen 'VICTIMAS DEL TERRORISMO'.

- **Problemas de Salud Mental:**

  Creación de Variable Objetivo: La columna 'problemas_salud_mental' muestra que ciertos tipos de matrícula están asociados con un mayor riesgo de problemas de salud mental, como 'Riesgo Académico' y 'Reincorporados Trikas'.

- **Modelo Predictivo:**

  Importancia de Características: Las características más importantes para predecir problemas de salud mental incluyen el tipo de matrícula y el ciclo relativo.

  Métricas del Modelo: El modelo de Random Forest ha mostrado un buen rendimiento en términos de precisión y AUC-ROC, indicando que es efectivo para predecir la salud mental de los estudiantes.

- **Visualización:**

  Importancia de Características: La visualización de la importancia de las características revela qué variables influyen más en las predicciones del modelo.

  Matriz de Confusión: La matriz de confusión muestra cómo el modelo clasifica los casos verdaderos y falsos, proporcionando información sobre la precisión de las predicciones.

Este análisis permite comprender mejor los patrones en los datos y la efectividad del modelo en la predicción de problemas de salud mental entre los estudiantes.