```
#Itzel Rubí Alcalá Gil clasificación de base de datos de pacientes con Parkinson


import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report



from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

```
df = pd.read_csv('/content/drive/MyDrive/Fotos/parkinsons_disease_data.csv')



print(df.dtypes)
print(df.info())
```

```
CholesterolHDL              float64
CholesterolTriglycerides    float64
UPDRS                       float64
MoCA                        float64
FunctionalAssessment        float64
```

```
 7   AlcoholConsumption        2105 non-null   float64
 8   PhysicalActivity          2105 non-null   float64
 9   DietQuality               2105 non-null   float64
 10  SleepQuality              2105 non-null   float64
 11  FamilyHistoryParkinsons   2105 non-null   int64
 12  TraumaticBrainInjury      2105 non-null   int64
 13  Hypertension              2105 non-null   int64
 14  Diabetes                  2105 non-null   int64
 15  Depression                2105 non-null   int64
 16  Stroke                    2105 non-null   int64
 17  SystolicBP                2105 non-null   int64
 18  DiastolicBP               2105 non-null   int64
 19  CholesterolTotal          2105 non-null   float64
 20  CholesterolLDL            2105 non-null   float64
 21  CholesterolHDL            2105 non-null   float64
 22  CholesterolTriglycerides  2105 non-null   float64
 23  UPDRS                     2105 non-null   float64
 24  MoCA                      2105 non-null   float64
 25  FunctionalAssessment      2105 non-null   float64
 26  Tremor                    2105 non-null   int64
 27  Rigidity                  2105 non-null   int64
 28  Bradykinesia              2105 non-null   int64
 29  PosturalInstability       2105 non-null   int64
 30  SpeechProblems            2105 non-null   int64
 31  SleepDisorders            2105 non-null   int64
 32  Constipation              2105 non-null   int64
 33  Diagnosis                 2105 non-null   int64
 34  DoctorInCharge            2105 non-null   object
dtypes: float64(12), int64(22), object(1)
memory usage: 575.7+ KB
None
```

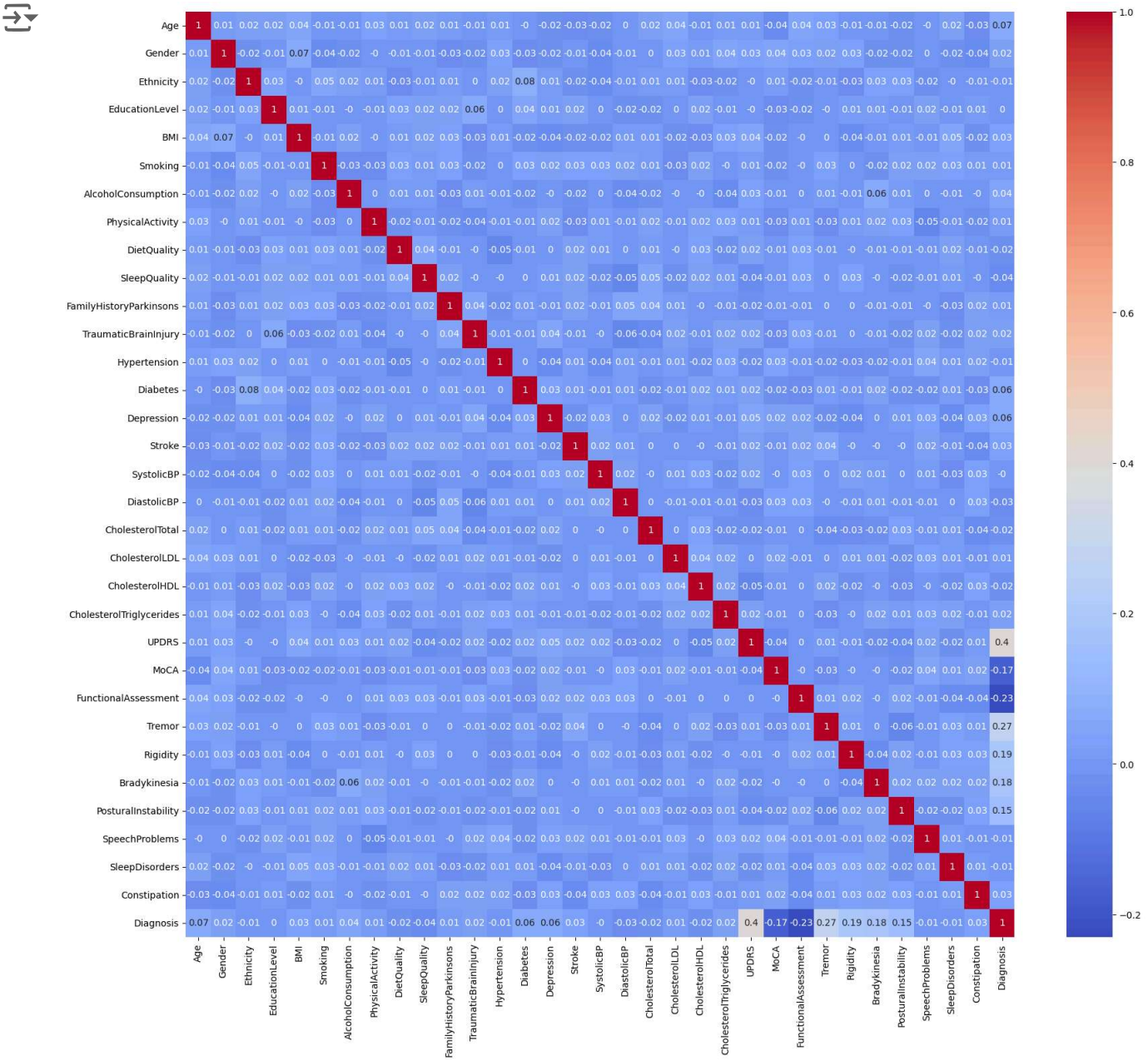
```
df=df.drop('DoctorInCharge',axis=1)
df=df.drop('PatientID',axis=1)
```

```
df.describe()
```

	Age	Gender	Ethnicity	EducationLevel	BMI	Smoking
count	2105.000000	2105.000000	2105.000000	2105.000000	2105.000000	2105.000000
mean	69.601900	0.492637	0.692637	1.337292	27.209493	0.296437
std	11.594511	0.500065	1.003827	0.895840	7.208099	0.456795
min	50.000000	0.000000	0.000000	0.000000	15.008333	0.000000
25%	60.000000	0.000000	0.000000	1.000000	20.782176	0.000000
50%	70.000000	0.000000	0.000000	1.000000	27.184571	0.000000
75%	80.000000	1.000000	1.000000	2.000000	33.462452	1.000000
max	89.000000	1.000000	3.000000	3.000000	39.999887	1.000000

8 rows × 33 columns

```
# Preparación de datos
#X = df[['Age', 'Gender','Ethnicity','EducationLevel','BMI','Smoking','AlcoholConsumption
#y = df['Diagnosis']
#X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42


# Matriz de correlación
plt.figure(figsize=(20, 18))
correlation_matrix = df.corr().round(2)
sns.heatmap(data=correlation_matrix, annot=True, cmap='coolwarm')
plt.show()
```


Age
Gender
Ethnicity
EducationLevel
BMI
Smoking
AlcoholConsumption
PhysicalActivity
DietQuality
SleepQuality
FamilyHistoryParkinsons
TraumaticBrainInjury
Hypertension
Diabetes
Depression
Stroke
SystolicBP
DiastolicBP
CholesterolTotal
CholesterolLDL
CholesterolHDL
CholesterolTriglycerides
UPDRS
MoCA
FunctionalAssessment
Tremor
Rigidity
Bradykinesia
PosturalInstability
SpeechProblems
SleepDisorders
Constipation
Diagnosis
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

```
#Creando un nuevo daata frame con las variables que tengan una correlación mayor a 15% co
nuevo_df=df[['UPDRS','MoCA','FunctionalAssessment','Tremor','Rigidity','Bradykinesia','Po
print(nuevo_df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2105 entries, 0 to 2104
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   UPDRS                 2105 non-null   float64
 1   MoCA                  2105 non-null   float64
 2   FunctionalAssessment  2105 non-null   float64
 3   Tremor                2105 non-null   int64  
 4   Rigidity              2105 non-null   int64  
 5   Bradykinesia          2105 non-null   int64  
 6   PosturalInstability   2105 non-null   int64  
 7   Diagnosis             2105 non-null   int64  
dtypes: float64(3), int64(5)
memory usage: 131.7 KB
None
```

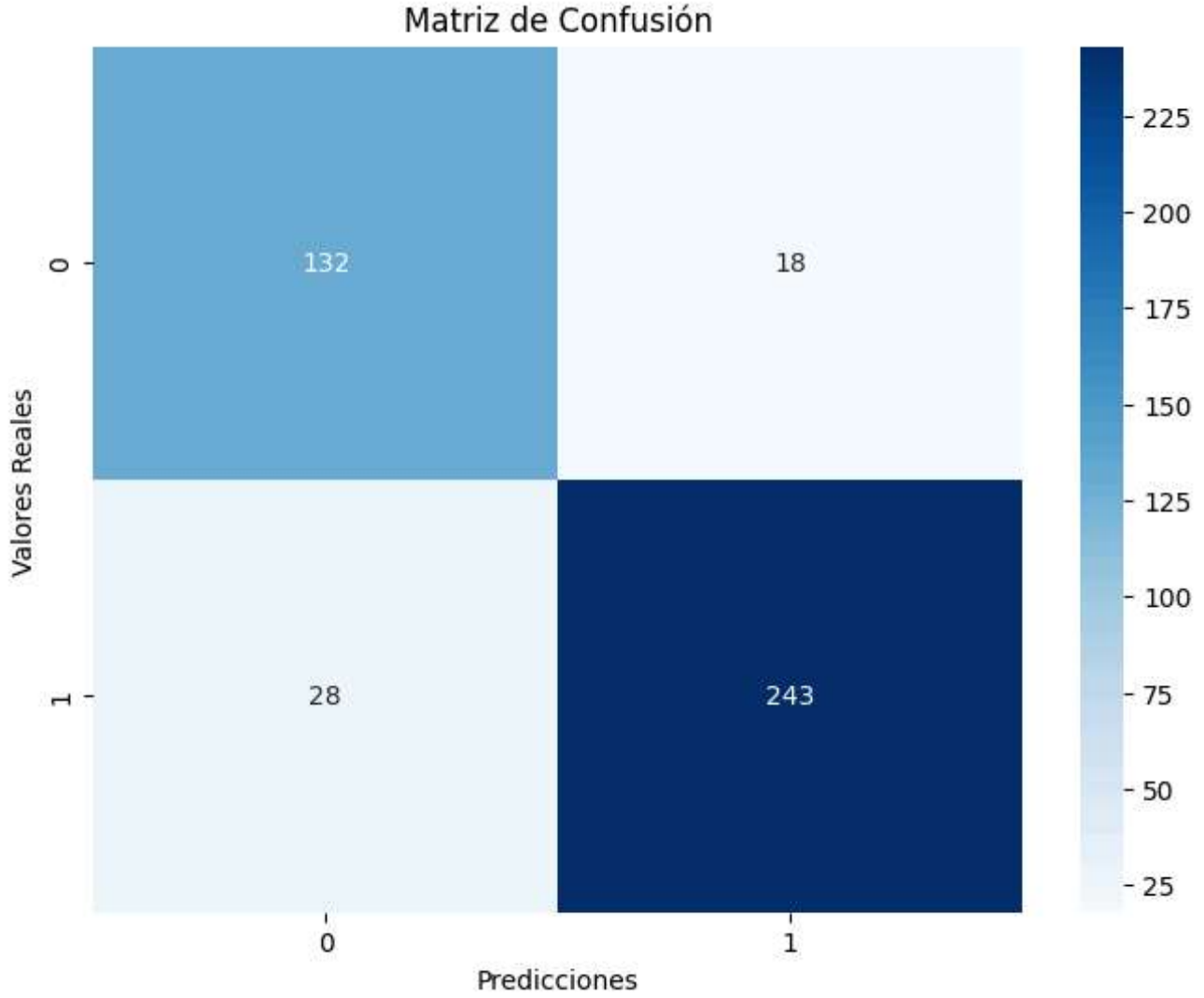
```
X = nuevo_df[['UPDRS','MoCA','FunctionalAssessment','Tremor','Rigidity','Bradykinesia','P
y = nuevo_df['Diagnosis']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Modelo Árbol de Decisión
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```

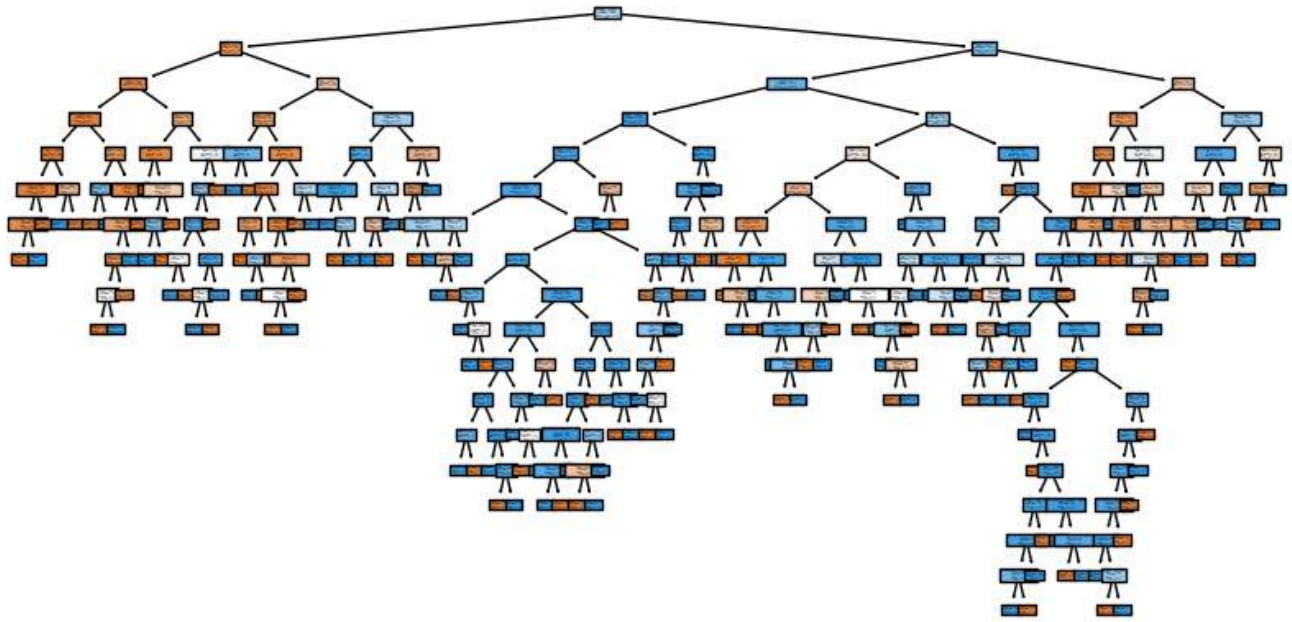
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
# Predicciones y matriz de confusión
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')

plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicciones')
plt.ylabel('Valores Reales')
plt.title('Matriz de Confusión')
plt.show()
```

```
Accuracy: 0.8907363420427553
```




```
# Visualización del árbol
plt.figure(figsize=(10,5))
plot_tree(model, feature_names=['UPDRS','MoCA','FunctionalAssessment','Tremor','Rigidity'
plt.show()
```



```
from sklearn.svm import SVC


# Entrenar el modelo SVM
model = SVC(kernel='rbf')
model.fit(X_train, y_train)
```

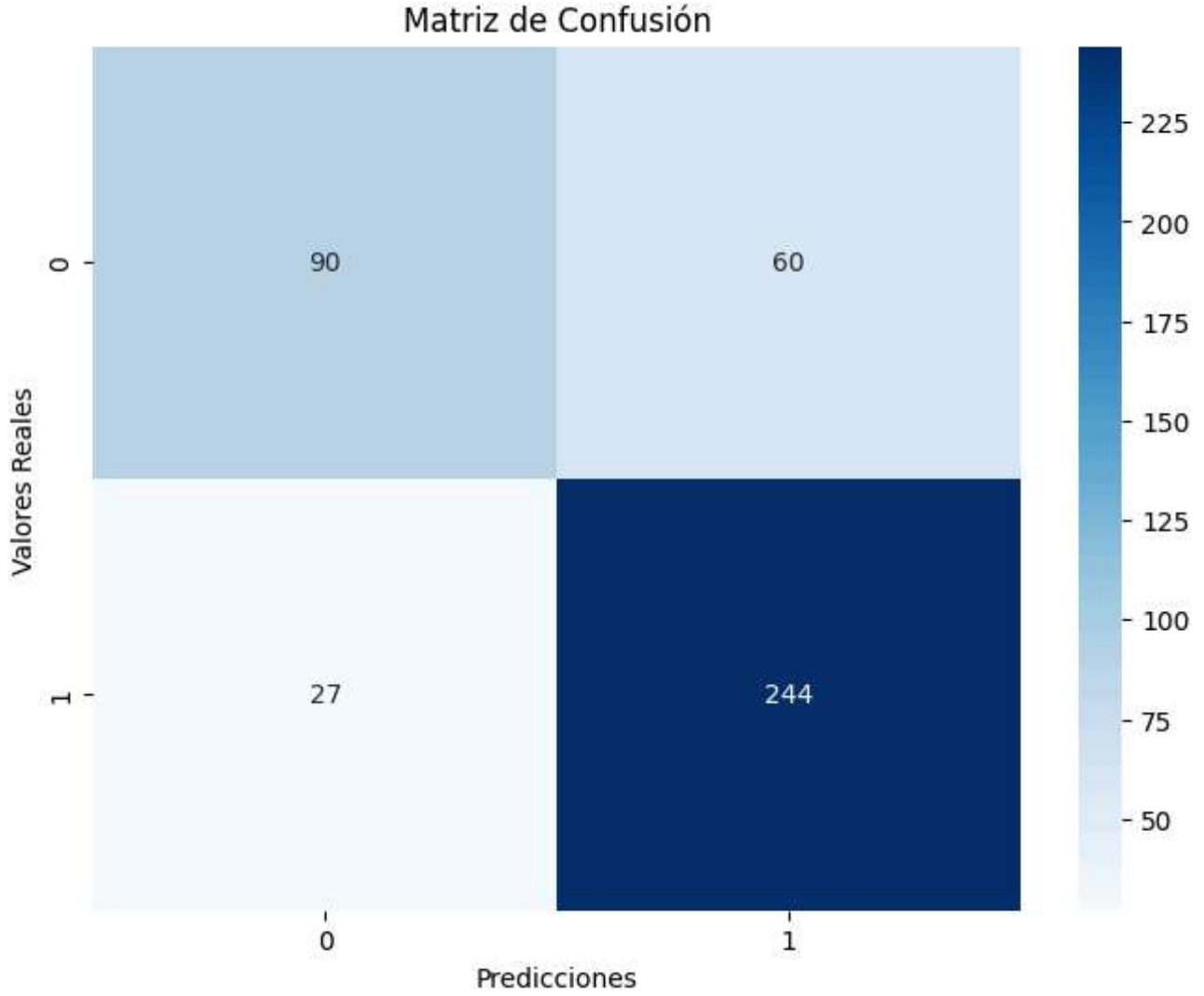
```
SVC()
```

```
# Predicciones
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')
```

```
Accuracy: 0.7933491686460807
```

```
# Predicciones y matriz de confusión
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')

plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicciones')
plt.ylabel('Valores Reales')
plt.title('Matriz de Confusión')
plt.show()
```

Accuracy: 0.7933491686460807




```
# modelo de regresión logística
model = LogisticRegression()

# Entrenar el modelo
model.fit(X_train, y_train)
```

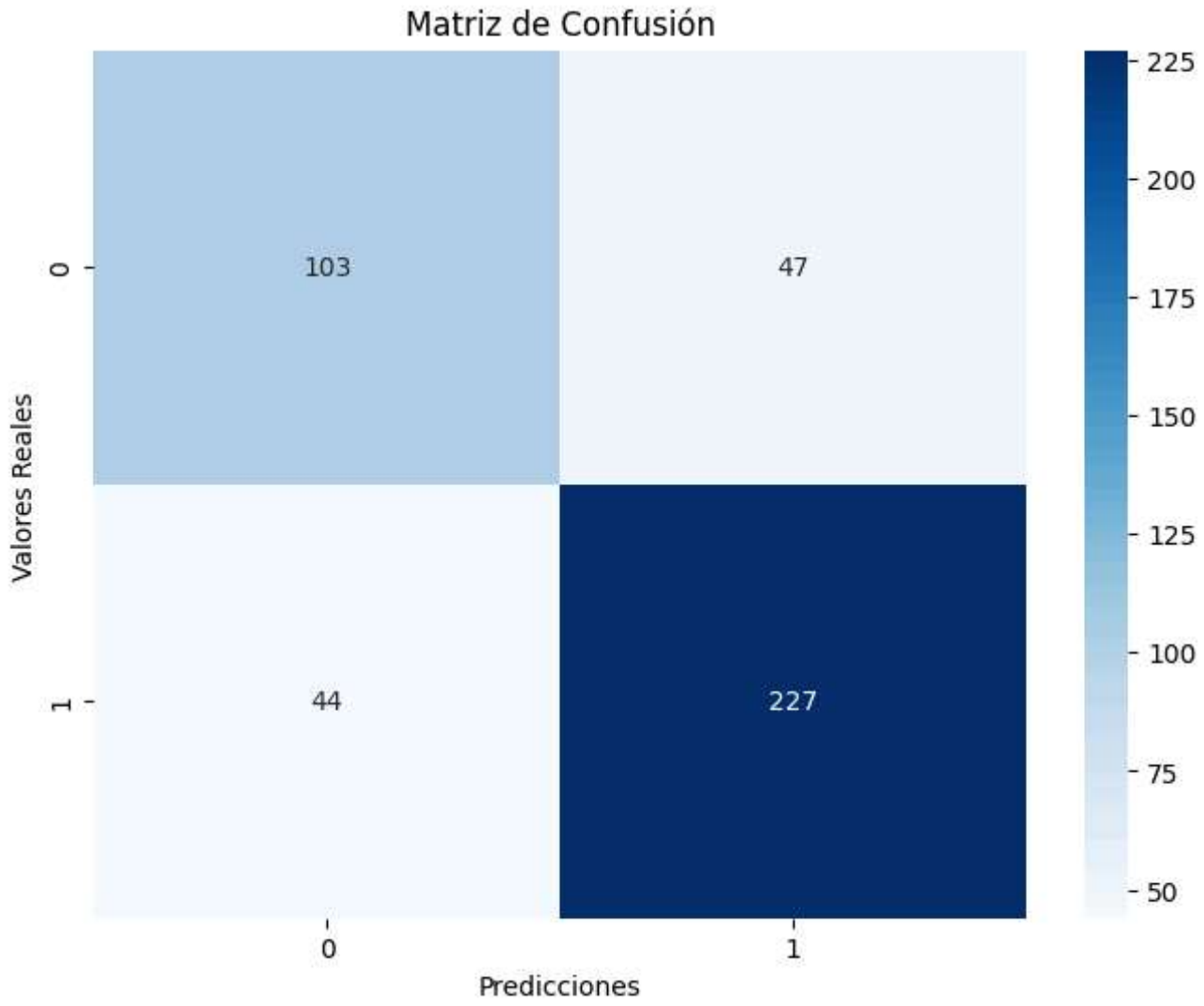
```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: Conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or 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(
```

▾ LogisticRegression

LogisticRegression()

```
# Predicciones
y_train_pred = model.predict(X_train)
y_test_pred = model.predict(X_test)

# Métricas de evaluación
print(f"Precisión en entrenamiento: {accuracy_score(y_train, y_train_pred)}")
print(f"Precisión en prueba: {accuracy_score(y_test, y_test_pred)}")
print("Informe de clasificación:")
print(classification_report(y_test, y_test_pred))

plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix(y_test, y_test_pred), annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicciones')
plt.ylabel('Valores Reales')
plt.title('Matriz de Confusión')
plt.show()
```

```
Precisión en entrenamiento: 0.8242280285035629
Precisión en prueba: 0.7838479809976246
Informe de clasificación:
              precision    recall  f1-score   support

           0       0.70      0.69      0.69       150
           1       0.83      0.84      0.83       271

    accuracy                           0.78       421
   macro avg       0.76      0.76      0.76       421
weighted avg       0.78      0.78      0.78       421
```



#hasta ahota el test que mejor acurracy me arrojó fue el de árboles de decisión con un 89

pip install lazypredict

```
Collecting lazypredict
  Downloading lazypredict-0.2.12-py2.py3-none-any.whl.metadata (12 kB)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: lightgbm in /usr/local/lib/python3.10/dist-packages (f
Requirement already satisfied: xgboost in /usr/local/lib/python3.10/dist-packages (fr
```

```
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/di
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (f
Downloading lazypredict-0.2.12-py2.py3-none-any.whl (12 kB)
Installing collected packages: lazypredict
Successfully installed lazypredict-0.2.12
WARNING: Running pip as the 'root' user can result in broken permissions and conflict
```

```
from lazypredict.Supervised import LazyClassifier


clf = LazyClassifier(verbose=0,ignore_warnings=True, custom_metric=None)
models,predictions = clf.fit(X_train, X_test, y_train, y_test)

print(models)
```

```
                               Accuracy  Balanced Accuracy  ROC AUC  F1 Score
AdaBoostClassifier                 0.93               0.92     0.92      0.93
ExtraTreesClassifier               0.91               0.91     0.91      0.91
LGBMClassifier                     0.91               0.91     0.91      0.91
XGBClassifier                      0.91               0.91     0.91      0.91
BaggingClassifier                  0.89               0.88     0.88      0.89
KNeighborsClassifier               0.89               0.88     0.88      0.89
LabelSpreading                     0.88               0.87     0.87      0.88
DecisionTreeClassifier             0.87               0.87     0.87      0.88
LabelPropagation                   0.87               0.87     0.87      0.87
SVC                                0.88               0.87     0.87      0.88
NuSVC                              0.84               0.82     0.82      0.84
ExtraTreeClassifier                0.82               0.81     0.81      0.82
QuadraticDiscriminantAnalysis      0.79               0.79     0.79      0.80
NearestCentroid                    0.78               0.79     0.79      0.79
LinearDiscriminantAnalysis         0.80               0.78     0.78      0.80
RidgeClassifier                    0.80               0.78     0.78      0.80
GaussianNB                         0.79               0.78     0.78      0.79
RidgeClassifierCV                  0.79               0.78     0.78      0.79
SGDClassifier                      0.78               0.77     0.77      0.79
CalibratedClassifierCV             0.78               0.76     0.76      0.78
LinearSVC                          0.78               0.76     0.76      0.78
LogisticRegression                 0.78               0.76     0.76      0.78
Perceptron                         0.75               0.75     0.75      0.75
PassiveAggressiveClassifier        0.73               0.73     0.73      0.73
BernoulliNB                        0.76               0.72     0.72      0.76
```

LabelSpreading	0.39
DecisionTreeClassifier	0.03
LabelPropagation	0.40
SVC	0.35
NuSVC	0.52
ExtraTreeClassifier	0.03
QuadraticDiscriminantAnalysis	0.04
NearestCentroid	0.08
LinearDiscriminantAnalysis	0.14