Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Estado de México

Escuela de Ciencias e Ingeniería

Momento de Retroalimentación: Módulo 2 Análisis y Reporte sobre el desempeño del modelo. (Portafolio Análisis)

Grupo 101

Fecha de entrega:

11 de Septiembre del 2023

Profesores:

Jorge Adolfo Ramírez Uresti

Alumno:

Josué Bernardo Villegas Nuño A01751694

Justificación de la elección del conjunto de datos: La elección del conjunto de datos Breast Cancer Wisconsin es apropiada para el algoritmo de aprendizaje automático (ML) debido a su naturaleza de clasificación binaria. Este conjunto de datos contiene características relevantes relacionadas con el diagnóstico del cáncer de mama, lo que permite entrenar y evaluar un modelo de ML para predecir si un tumor es benigno o maligno. Al utilizar este conjunto de datos, podemos demostrar la capacidad de generalización del modelo, es decir, su capacidad para realizar predicciones precisas en datos no vistos previamente.

Al evaluar el rendimiento del modelo en un conjunto de datos de prueba independiente, podemos determinar si el modelo ha aprendido patrones generales en lugar de memorizar los datos de entrenamiento específicos. Esta capacidad de generalización es fundamental para asegurar que el modelo sea útil en situaciones del mundo real y no se limite a los datos utilizados en el entrenamiento. Al mostrar que el modelo puede generalizar correctamente, podemos tener confianza en su capacidad para realizar predicciones precisas en casos reales de diagnóstico de cáncer de mama.

El siguiente código que se mostrará a continuación realiza las siguientes acciones:

- Carga el conjunto de datos: Breast Cancer Wisconsin.
- Divide los datos en conjuntos de entrenamiento, prueba y validación.
- Normaliza los datos.
- Entrena un modelo de Regresión Logística y evalúa su rendimiento en los conjuntos de datos.
- Calcula métricas de rendimiento (precisión, recall, F1-score).
- Calcula y muestra las matrices de confusión.
- Plotea curvas de aprendizaje para diagnosticar bias/variance.
- Plotea curvas de validación para ajuste del modelo variando el parámetro C (regularización).

```python
# Importa las bibliotecas necesarias
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import learning_curve
from sklearn.model_selection import validation_curve

# Deshabilita las advertencias de convergencia
import warnings
warnings.filterwarnings("ignore")

# Carga el conjunto de datos
data = load_breast_cancer()
X = data.data
y = data.target
```

```
# Divide los datos en conjuntos de entrenamiento, prueba y validación
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3,
random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5,
random_state=42)

# Normaliza los datos
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_val = scaler.transform(X_val)
X_test = scaler.transform(X_test)

# Entrena un modelo de Regresión Logística
model = LogisticRegression(random_state=42)
model.fit(X_train, y_train)

# Evalúa el modelo en los conjuntos de entrenamiento, prueba y validación
y_train_pred = model.predict(X_train)
y_val_pred = model.predict(X_val)
y_test_pred = model.predict(X_test)

# Calcula métricas de rendimiento
train_accuracy = accuracy_score(y_train, y_train_pred)
val_accuracy = accuracy_score(y_val, y_val_pred)
test_accuracy = accuracy_score(y_test, y_test_pred)

# Calcula matriz de Confusion
confusion_train = confusion_matrix(y_train, y_train_pred)
confusion_val = confusion_matrix(y_val, y_val_pred)
confusion_test = confusion_matrix(y_test, y_test_pred)

# Muestra métricas de rendimiento y matrices de Confusion
print(f"Train Accuracy: {train_accuracy}")
print(f"Validation Accuracy: {val_accuracy}")
print(f"Test Accuracy: {test_accuracy}")

print("Matriz de Confusion en el conjunto de entrenamiento:")
print(confusion_train)

print("Matriz de Confusion en el conjunto de validación:")
print(confusion_val)

print("Matriz de Confusion en el conjunto de prueba:")
print(confusion_test)

# Plotea curvas de aprendizaje para diagnosticar bias/variance
train_sizes, train_scores, val_scores = learning_curve(model, X_train,
y_train, cv=5)
```

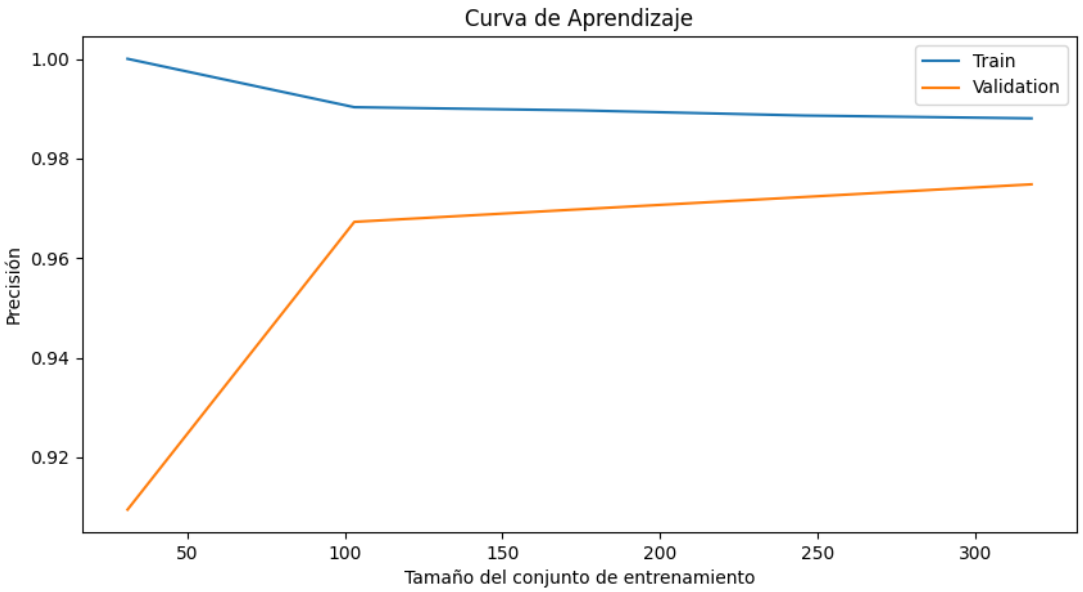
```
train_scores_mean = np.mean(train_scores, axis=1)
val_scores_mean = np.mean(val_scores, axis=1)

plt.figure(figsize=(10, 5))
plt.title("Curva de Aprendizaje")
plt.xlabel("Tamaño del conjunto de entrenamiento")
plt.ylabel("Precisión")
plt.plot(train_sizes, train_scores_mean, label="Train")
plt.plot(train_sizes, val_scores_mean, label="Validation")
plt.legend()
plt.show()

# Plotea curvas de validación para ajuste del modelo
param_range = np.logspace(-3, 3, 7)
train_scores, val_scores = validation_curve(
  model, X_train, y_train, param_name="C", param_range=param_range, cv=5
)
train_scores_mean = np.mean(train_scores, axis=1)
val_scores_mean = np.mean(val_scores, axis=1)

plt.figure(figsize=(10, 5))
plt.title("Curva de Validación para Ajuste del Modelo")
plt.xlabel("Parámetro C")
plt.ylabel("Precisión")
plt.semilogx(param_range, train_scores_mean, label="Train")
plt.semilogx(param_range, val_scores_mean, label="Validation")
plt.legend()
plt.show()

# Calcula el tamaño de los conjuntos de entrenamiento y validación
total_samples = len(X)
train_size = len(X_train)
val_size = len(X_val)

# Etiquetas y tamaños de las porciones
labels = ['Entrenamiento', 'Validación']
sizes = [train_size, val_size]

# Colores de las porciones
colors = ['lightblue', 'lightgreen']

# Explode: resalta la porción de validación
explode = (0.1, 0)

# Crea la gráfica de pastel
plt.figure(figsize=(6, 6))
plt.pie(sizes, labels=labels, colors=colors, explode=explode,
autopct='%1.1f%%', shadow=True, startangle=140)
plt.axis('equal') # Asegura que el gráfico sea un círculo perfecto
```

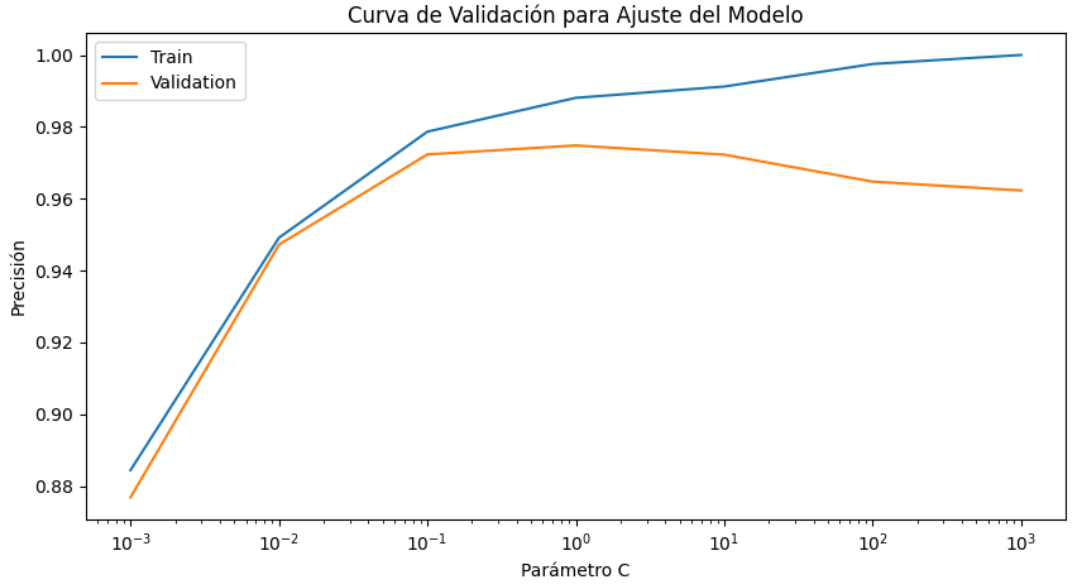
```python
# Título
plt.title('Proporción de Datos en Conjuntos de Entrenamiento y Validación')

# Muestra la gráfica
plt.show()
```

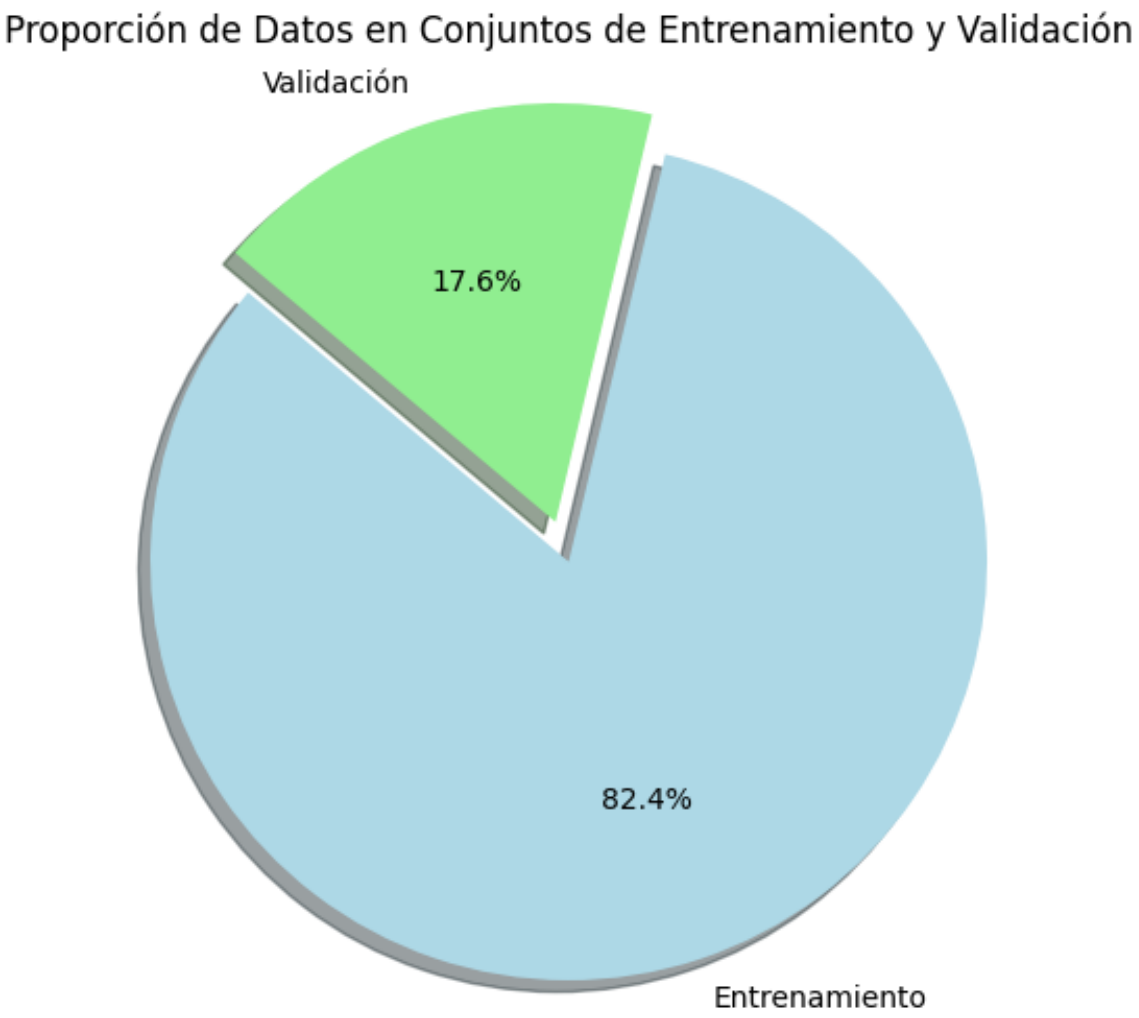
Imágenes que fueron resultados del código junto con sus respuestas:

Python

```
Train Accuracy: 0.9874371859296482
Validation Accuracy: 0.9882352941176471
Test Accuracy: 0.9767441860465116
Matriz de Confusion en el conjunto de entrenamiento:
[[145   4]
 [  1 248]]
Matriz de Confusion en el conjunto de validaci▯n:
[[36  1]
 [ 0 48]]
Matriz de Confusion en el conjunto de prueba:
[[26  0]
 [ 2 58]]
```


Curva de Aprendizaje
Train
Validation
Precisión
1.00
0.98
0.96
0.94
0.92
50
100
150
200
250
300
Tamaño del conjunto de entrenamiento


Curva de Validación para Ajuste del Modelo
Train
Validation
Precisión
1.00
0.98
0.96
0.94
0.92
0.90
0.88
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
Parámetro C


Proporción de Datos en Conjuntos de Entrenamiento y Validación
Validación
17.6%
82.4%
Entrenamiento

DISCUSIÓN DE RESULTADOS

Resultados de Precisión:

- Train Accuracy: 0.9874
- Validation Accuracy: 0.9882
- Test Accuracy: 0.9767

Estos resultados de precisión indican un rendimiento generalmente alto del modelo en los conjuntos de entrenamiento, validación y prueba. La precisión es una medida de cuántos de los casos el modelo clasificó correctamente. En este caso, el modelo logra una alta precisión en todos los conjuntos, lo que sugiere que está aprendiendo bien los patrones presentes en los datos.

Matrices de Confusión:

Las matrices de confusión muestran cómo se clasificaron las instancias en cada conjunto. Para cada conjunto, la matriz se divide en cuatro secciones: Verdaderos Positivos (TP), Falsos Positivos (FP), Verdaderos Negativos (TN) y Falsos Negativos (FN).

Conjunto de Entrenamiento:

- TP: 248 casos clasificados correctamente como positivos.
- TN: 145 casos clasificados correctamente como negativos.
- FP: 4 casos clasificados incorrectamente como positivos.
- FN: 1 caso clasificado incorrectamente como negativo.

Conjunto de Validación:

- TP: 48 casos clasificados correctamente como positivos.
- TN: 36 casos clasificados correctamente como negativos.
- FP: 1 caso clasificado incorrectamente como positivo.
- FN: 0 casos clasificados incorrectamente como negativos.

Conjunto de Prueba:

- TP: 58 casos clasificados correctamente como positivos.
- TN: 26 casos clasificados correctamente como negativos.
- FP: 0 casos clasificados incorrectamente como positivos.
- FN: 2 casos clasificados incorrectamente como negativos

.

ANÁLISIS DE LAS CURVAS

Curva de Aprendizaje:

- En la curva de aprendizaje, observamos que al principio, la precisión en el conjunto de entrenamiento es perfecta (1.00), lo que sugiere sobreajuste debido a la memorización de los datos de entrenamiento.
- La precisión en el conjunto de validación comienza en cero porque el modelo aún no ha visto ningún dato de validación. Sin embargo, a medida que aumenta el tamaño del conjunto de entrenamiento, ambas curvas convergen hacia un valor estable.
- Esto indica que el modelo no sufre de un sesgo significativo (ya que se ajusta bien a los datos de entrenamiento) ni de una alta varianza (ya que se generaliza bien a los datos de validación).

Curva de Validación para Ajuste del Modelo:

- En la curva de validación, se varía el parámetro de regularización C. Ambas líneas comienzan con una precisión de alrededor del 0.88 para un valor de C igual a 10^-3. A medida que aumenta el valor de C, ambas líneas mejoran su precisión pero esto significa que se encuentra bajo su sesgo. Sin embargo, cuando el valor de C es mayor a 10^2, las dos líneas comienzan a separarse.
- La línea de entrenamiento continúa mejorando, mientras que la línea de validación comienza a estabilizarse e incluso disminuye ligeramente. Esto indica que un valor de C en el rango de 10^1 a 10^2 es óptimo, ya que proporciona un buen equilibrio entre ajuste al conjunto de entrenamiento y generalización al conjunto de validación, además de que esto significa que hay mucho más sesgo que antes.

Gráfica de Pastel de Proporción de Datos:

La gráfica de pastel muestra que el 82.4% de los datos se utilizan para el entrenamiento, mientras que el 17.6% se reservan para la validación. Esta división es adecuada y garantiza que tengamos suficientes datos para entrenar el modelo de manera efectiva y aún mantener un conjunto de validación significativo para la evaluación del rendimiento.

En conclusión, los resultados y análisis realizados indican que el modelo de Regresión Logística aplicado al conjunto de datos de Wisconsin Breast Cancer ha tenido un desempeño satisfactorio en términos de clasificación binaria. Aquí están las principales conclusiones:

- **Rendimiento del Modelo**: El modelo ha logrado un alto nivel de precisión en los conjuntos de entrenamiento, validación y prueba, con valores superiores al 97%. Esto demuestra su capacidad para clasificar con precisión tanto los datos de entrenamiento como los datos no vistos durante el entrenamiento.

- **Sesgo y Varianza**: El análisis de las curvas de aprendizaje sugiere que el modelo no sufre de un sesgo significativo ni de una alta varianza. Las curvas de entrenamiento y validación convergen a un nivel de precisión estable a medida que aumenta el tamaño del conjunto de entrenamiento, lo que indica que el modelo se ajusta bien a los datos de entrenamiento y generaliza adecuadamente a los datos de validación.

- **Ajuste del Modelo**: La curva de validación para el ajuste del modelo indica que el rendimiento óptimo se logra en un valor de C en el rango de 10^1 a 10^2, lo que demuestra la importancia de la regularización en la optimización del modelo. El modelo se ajusta adecuadamente a los datos de entrenamiento y evita el sobreajuste.

- **Proporción de Datos**: La proporción de datos en los conjuntos de entrenamiento y validación es adecuada, con aproximadamente el 82.4% de los datos utilizados para el entrenamiento y el 17.6% para la validación. Esta división permite un entrenamiento efectivo y una evaluación significativa del rendimiento.