Avance3_36

May 19, 2024

1 Proyecto Integrador

1.1 Score de Riesgo en Originación de Crédito

1.2 Para Kubo Financiero

Participantes:

Dalina Aideé Villa Ocelotl (A01793258)

Julián Valera Juarez (A01793875)

Miguel Guillermo Galindo Orozco (A01793695) 1793695)

1.3 Avance 3. Baseline

[1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, classification_report, roc_curve,␣
 ↪auc, accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
# Leer archivo Excel
file_path = 'base_20210101a20240430 (2).xlsx'
df = pd.read_excel(file_path, sheet_name='Cliientes_Nuevos')
```

[3]:
```
# Seleccionar las características y la variable objetivo
X = df.drop('es_malo_actual', axis=1)
y = df['es_malo_actual']
```

[4]:
```
# Rellenar valores nulos en columnas numéricas con la mediana
numeric_columns = X.select_dtypes(include=[np.number]).columns
for col in numeric_columns:
    median_value = X[col].median()
    X[col] = X[col].fillna(median_value)
```

[5]:
```
# Eliminar columnas tipo datetime
X = X.select_dtypes(exclude=['datetime64'])
```

[6]:
```
# Convertir características categóricas a variables dummy
X = pd.get_dummies(X, drop_first=True)
```

[7]:
```
# Escalar los datos
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

[8]:
```
# Dividir los datos en conjunto de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,␣
 ↪random_state=42)
```

[9]:
```
# Entrenar el modelo de regresión logística como baseline
model = LogisticRegression(max_iter=2000)
model.fit(X_train, y_train)
```

[9]:
```
LogisticRegression(max_iter=2000)
```

[10]:
```
# Hacer predicciones en el conjunto de prueba
y_pred = model.predict(X_test)
```

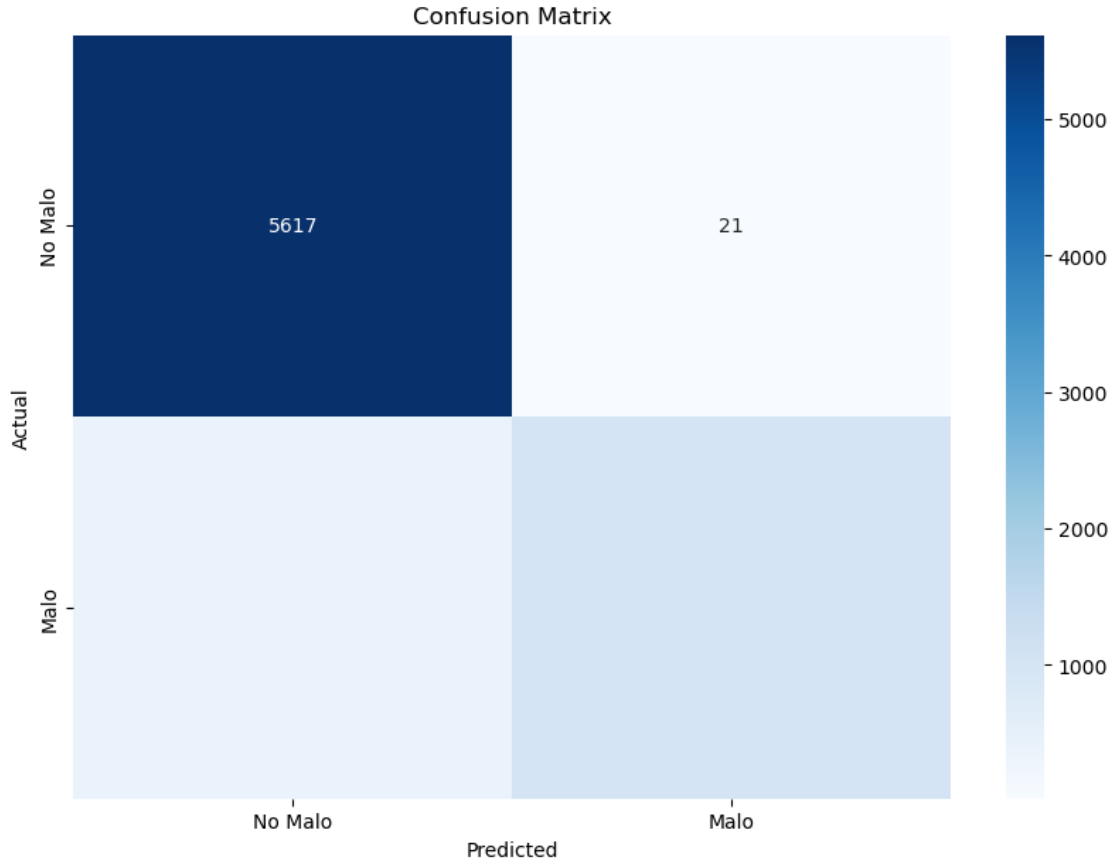
[11]:
```
# Calcular la precisión del modelo (Baseline)
accuracy = accuracy_score(y_test, y_pred)
print("Precisión del modelo (Baseline):", accuracy)
```

```
Precisión del modelo (Baseline): 0.9431818181818182
```

[12]:
```
# Crear Matriz de confusión
conf_matrix = confusion_matrix(y_test, y_pred)
print("Matriz de Confusión:")
print(conf_matrix)
```

```
Matriz de Confusión:
[[5617   21]
 [ 379 1023]]
```

[13]:
```
# Grafica de Matriz de confusión
plt.figure(figsize=(10, 7))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['No␣
 ↪Malo', 'Malo'], yticklabels=['No Malo', 'Malo'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



[14]:
```
# Informe de clasificación
class_report = classification_report(y_test, y_pred)
print("Informe de Clasificación:")
print(class_report)
```

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

           0       0.94      1.00      0.97      5638
           1       0.98      0.73      0.84      1402

    accuracy                           0.94      7040
   macro avg       0.96      0.86      0.90      7040
weighted avg       0.95      0.94      0.94      7040
```

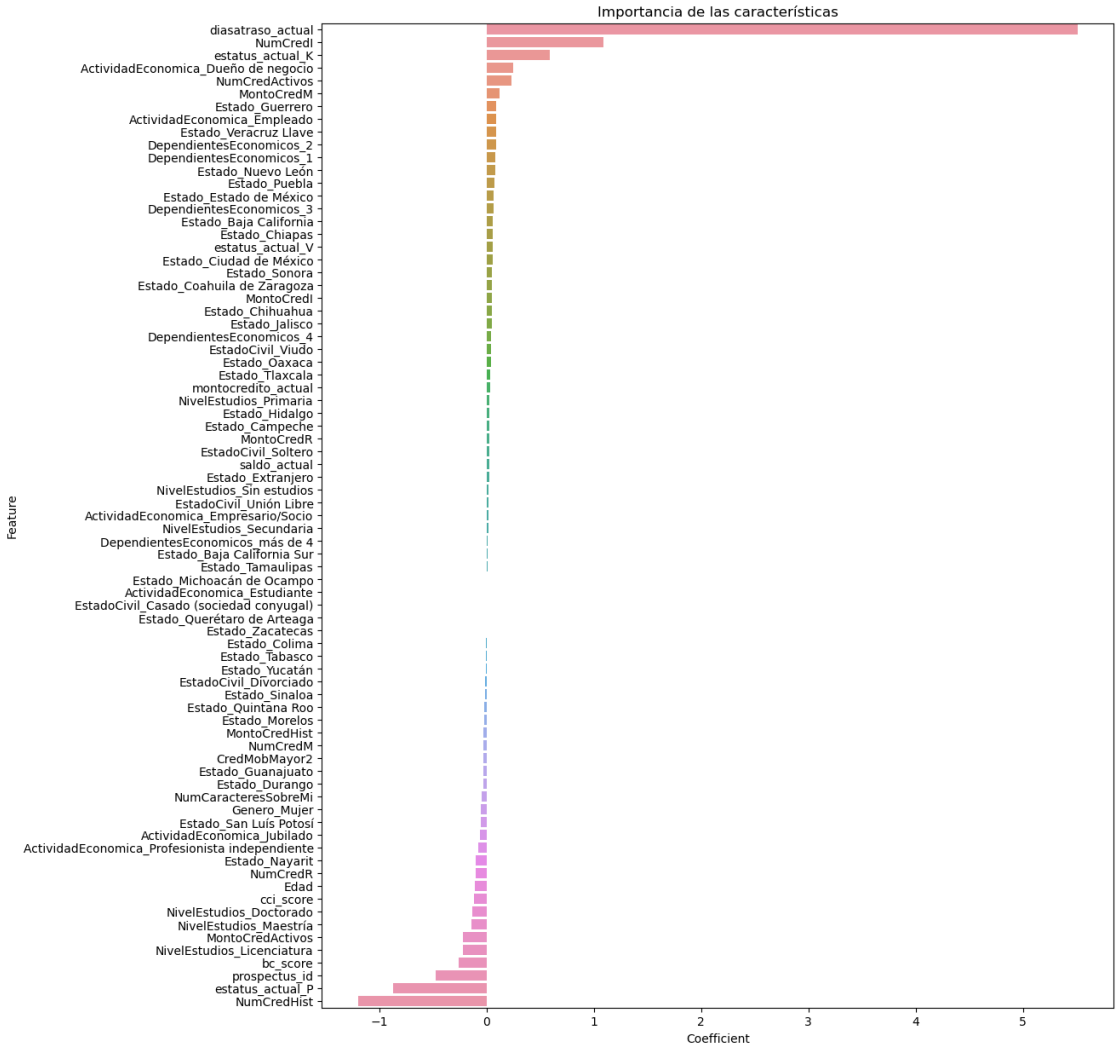
[15]:
```
# Identificar características importantes utilizando coeficientes de regresión␣
↪logística
feature_importance = pd.DataFrame({'Feature': X.columns, 'Coefficient': model.
↪coef_[0]})
```

```
print('Importancia de las características:')
pd.set_option('display.max_rows', None)
print(feature_importance)
```

```
Importancia de las características:
                                        Feature  Coefficient
0                                   prospectus_id    -0.476075
1                                            Edad    -0.107702
2                              NumCaracteresSobreMi    -0.047896
3                                       cci_score    -0.115357
4                                        bc_score    -0.261510
5                                     NumCredHist    -1.199730
6                                    MontoCredHist    -0.026729
7                                    CredMobMayor2    -0.028172
8                                    NumCredActivos     0.228527
9                                  MontoCredActivos    -0.218138
10                                         NumCredR    -0.104592
11                                       MontoCredR     0.027143
12                                         NumCredI     1.088303
13                                       MontoCredI     0.046397
14                                         NumCredM    -0.027485
15                                       MontoCredM     0.122739
16                              montocredito_actual     0.033903
17                                diasatraso_actual     5.511364
18                                     saldo_actual     0.023345
19                                     Genero_Mujer    -0.055482
20          EstadoCivil_Casado (sociedad conyugal)    -0.000777
21                           EstadoCivil_Divorciado    -0.011721
22                              EstadoCivil_Soltero     0.025604
23                           EstadoCivil_Unión Libre     0.016646
24                                 EstadoCivil_Viudo     0.041345
25                          DependientesEconomicos_1     0.080056
26                          DependientesEconomicos_2     0.085603
27                          DependientesEconomicos_3     0.062674
28                          DependientesEconomicos_4     0.044241
29                  DependientesEconomicos_más de 4     0.009540
30                           Estado_Baja California     0.058189
31                       Estado_Baja California Sur     0.006640
32                                Estado_Campeche     0.027213
33                                 Estado_Chiapas     0.055031
34                                Estado_Chihuahua     0.046226
35                           Estado_Ciudad de México     0.053871
36                       Estado_Coahuila de Zaragoza     0.048495
37                                  Estado_Colima    -0.002691
38                                 Estado_Durango    -0.032564
39                           Estado_Estado de México     0.063208
40                                Estado_Extranjero     0.022383
41                                Estado_Guanajuato    -0.029913
```

```
42                                  Estado_Guerrero     0.089579
43                                   Estado_Hidalgo     0.027800
44                                   Estado_Jalisco     0.045804
45                       Estado_Michoacán de Ocampo     0.005007
46                                   Estado_Morelos    -0.025938
47                                   Estado_Nayarit    -0.099215
48                                Estado_Nuevo León     0.078680
49                                    Estado_Oaxaca     0.038142
50                                    Estado_Puebla     0.074586
51                      Estado_Querétaro de Arteaga    -0.002015
52                              Estado_Quintana Roo    -0.021119
53                           Estado_San Luís Potosí    -0.055978
54                                   Estado_Sinaloa    -0.017215
55                                    Estado_Sonora     0.053004
56                                   Estado_Tabasco    -0.003022
57                                Estado_Tamaulipas     0.006083
58                                  Estado_Tlaxcala     0.035110
59                               Estado_Veracruz Llave     0.086130
60                                   Estado_Yucatán    -0.009883
61                                Estado_Zacatecas    -0.002155
62             ActividadEconomica_Dueño de negocio     0.249300
63                     ActividadEconomica_Empleado     0.088015
64             ActividadEconomica_Empresario/Socio     0.016633
65                   ActividadEconomica_Estudiante     0.002664
66                    ActividadEconomica_Jubilado    -0.062206
67  ActividadEconomica_Profesionista independiente    -0.081405
68                          NivelEstudios_Doctorado    -0.133442
69                       NivelEstudios_Licenciatura    -0.218432
70                           NivelEstudios_Maestría    -0.139884
71                           NivelEstudios_Primaria     0.028368
72                         NivelEstudios_Secundaria     0.015762
73                       NivelEstudios_Sin estudios     0.020815
74                                 estatus_actual_K     0.592170
75                                 estatus_actual_P    -0.874161
76                                 estatus_actual_V     0.053913
```

[16]:
```python
# Grafica de la importancia de las características
plt.figure(figsize=(12, 15))
sns.barplot(x='Coefficient', y='Feature', data=feature_importance.
 ↪sort_values(by='Coefficient', ascending=False))
plt.title('Importancia de las características')
plt.show()
```

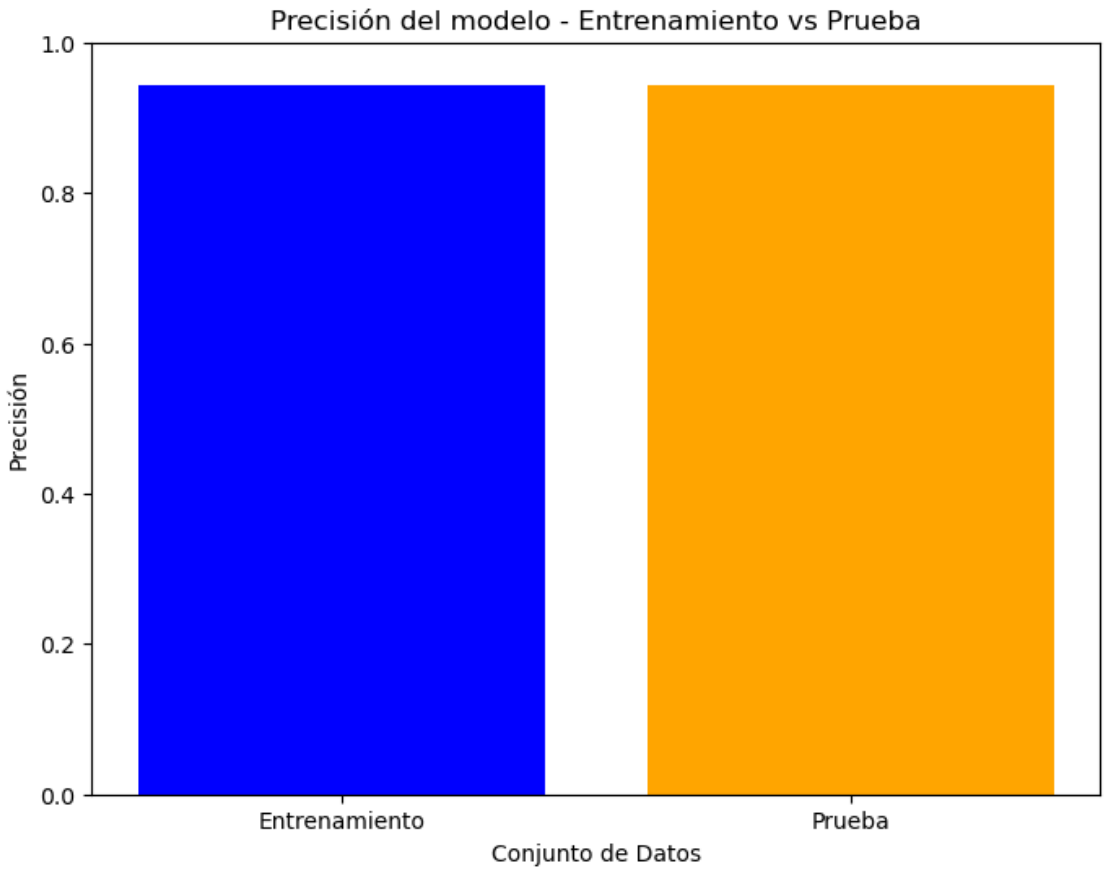


[17]:
```python
# Evaluar sobreajuste del modelo
train_score = model.score(X_train, y_train)
test_score = model.score(X_test, y_test)
print("Precisión del conjunto de entrenamiento:", train_score)
print("Precisión del conjunto de prueba:", test_score)
```

```
Precisión del conjunto de entrenamiento: 0.9431047341691231
Precisión del conjunto de prueba: 0.9431818181818182
```

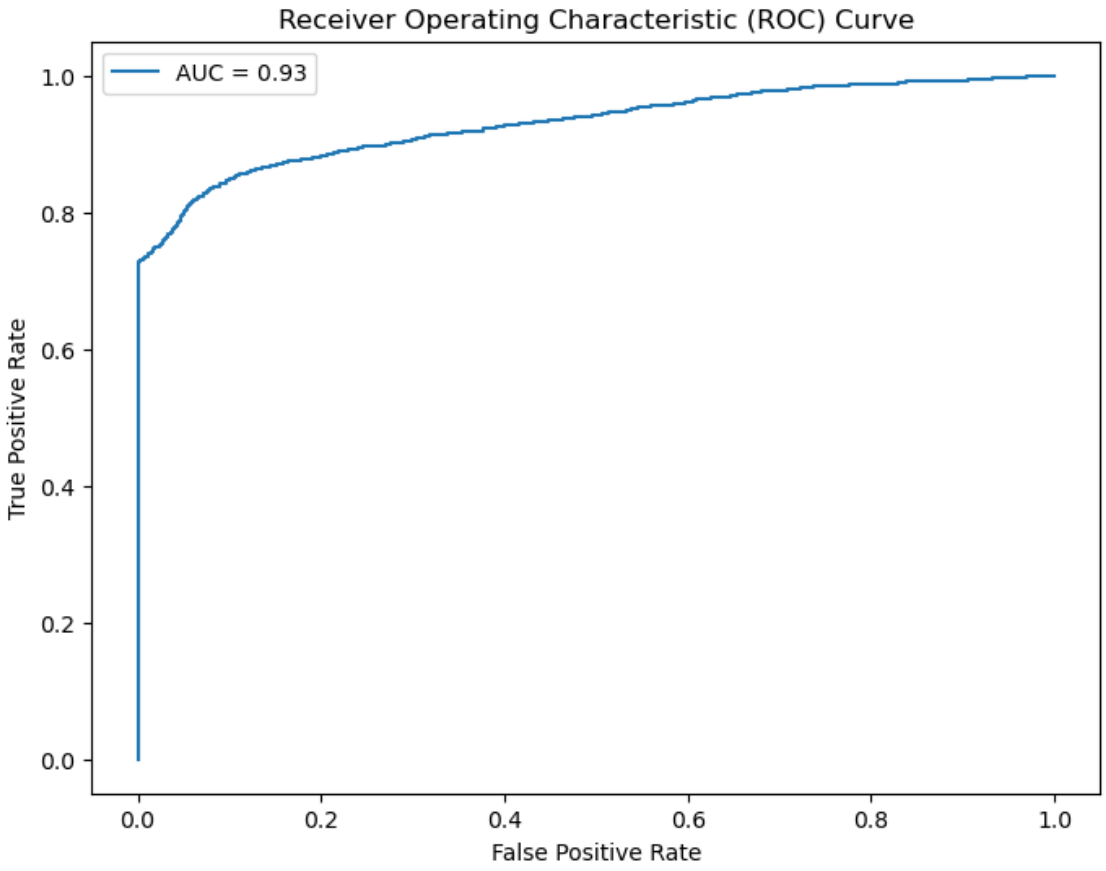
[18]:
```python
import matplotlib.pyplot as plt

# Definir métricas a graficar
metrics = ['Entrenamiento', 'Prueba']
accuracy_scores = [train_score, test_score]
```

```
# Graficar las métricas
plt.figure(figsize=(8, 6))
plt.bar(metrics, accuracy_scores, color=['blue', 'orange'])
plt.title('Precisión del modelo - Entrenamiento vs Prueba')
plt.xlabel('Conjunto de Datos')
plt.ylabel('Precisión')
plt.ylim(0, 1)
plt.show()
```



[19]:
```
# Calcular la curva ROC y AUC-ROC
y_probs = model.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_probs)
auc_score = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label=f'AUC = {auc_score:.2f}')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
```

```
plt.legend()
plt.show()
```



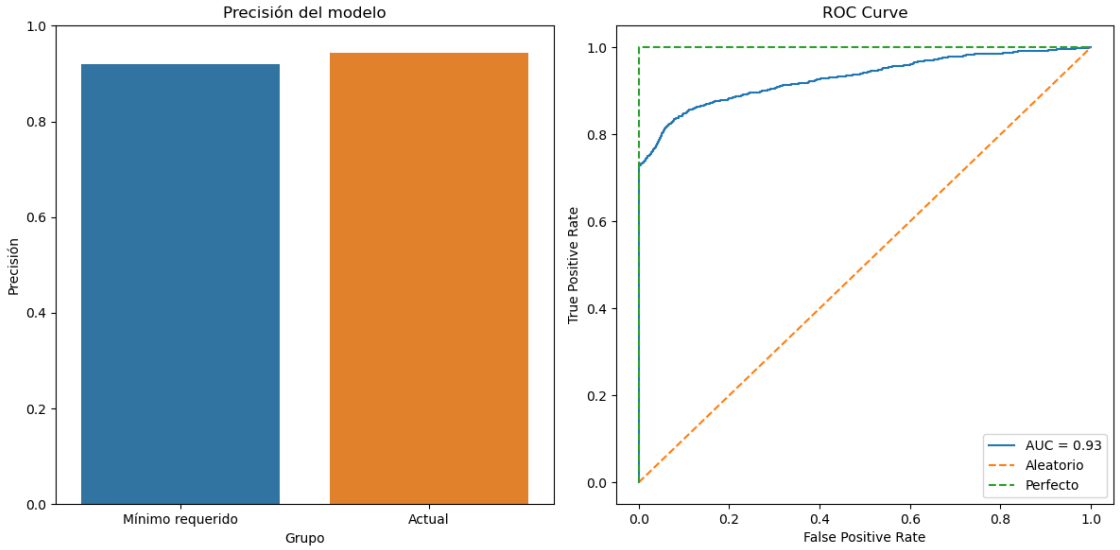
[20]:
```
# Establecer el desempeño mínimo a obtener
min_accuracy = 0.92
min_auc_roc = 0.90
min_recall = 0.70
min_precision = 0.95
min_f1_score = 0.80
min_specificity = 0.95

# Calcular otras métricas para comparar con el desempeño mínimo
report = classification_report(y_test, y_pred, output_dict=True)
recall = report['1']['recall']
precision = report['1']['precision']
f1_score = report['1']['f1-score']
specificity = report['0']['recall']  # Specificity is the recall of the␣
 ↪negative class
```

```
# Mostrar las métricas calculadas
print(f'Desempeño mínimo requerido - Accuracy: {min_accuracy}, AUC-ROC:␣
 ↪{min_auc_roc}')
print(f'Desempeño del modelo:\n   Accuracy: {accuracy},\n    AUC-ROC:␣
 ↪{auc_score},\n     Recall: {recall}, \n  Precision: {precision}, \n  ␣
 ↪F1-Score: {f1_score},\nSpecificity: {specificity}')

# Verificar si el modelo cumple con los requisitos mínimos
if (accuracy >= min_accuracy and auc_score >= min_auc_roc and
    recall >= min_recall and precision >= min_precision and
    f1_score >= min_f1_score and specificity >= min_specificity):
    print("\nEl modelo cumple con el desempeño mínimo requerido.")
else:
    print("\nEl modelo no cumple con el desempeño mínimo requerido.")
```

```
Desempeño mínimo requerido - Accuracy: 0.92, AUC-ROC: 0.9
Desempeño del modelo:
   Accuracy: 0.9431818181818182,
    AUC-ROC: 0.9300211677535614,
     Recall: 0.7296718972895863,
  Precision: 0.9798850574712644,
   F1-Score: 0.8364677023712183,
Specificity: 0.9962752749201844

El modelo cumple con el desempeño mínimo requerido.
```

[21]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Crear un DataFrame con los datos de precisión mínima y precisión actual
data = {'Grupo': ['Mínimo requerido', 'Actual'],
        'Precisión': [min_accuracy, accuracy]}
df = pd.DataFrame(data)

# Graficar usando Seaborn
plt.figure(figsize=(12, 6))

# Gráfico de precisión
plt.subplot(1, 2, 1)
sns.barplot(x='Grupo', y='Precisión', data=df)
plt.title('Precisión del modelo')
plt.ylabel('Precisión')
plt.ylim(0, 1)

# Gráfico de AUC-ROC
plt.subplot(1, 2, 2)
```

```
plt.plot(fpr, tpr, label=f'AUC = {auc_score:.2f}')
plt.plot([0, 1], [0, 1], linestyle='--', label='Aleatorio')
plt.plot([0, 0, 1], [0, 1, 1], linestyle='--', label='Perfecto')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()

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



1.4 ¿Qué algoritmo se puede utilizar como baseline para predecir las variables objetivo?

1.4.1 El algoritmo utilizado como baseline para predecir las variables es el de Regresión Logística.

1.4.2 La regresión logística es una opción apropiada para el Score de Riesgo en Originación de Crédito debido a su capacidad para proporcionar interpretación, manejar problemas binarios, eficiencia computacional y capacidad predictiva adecuada.

1.5 ¿Se puede determinar la importancia de las características para el modelo generado?

1.5.1 Sí, se puede determinar la importancia de las características utilizando los coeficientes del modelo de regresión logística. En el archivo se muestra cómo se crean las tablas con las características y sus coeficientes, lo que permite identificar cuáles son las más importantes.

1.6 ¿El modelo está sub/sobreajustando los datos de entrenamiento?

1.6.1 El modelo no está ni sobreajustando ni subajustando los datos de entrenamiento. Esto se debe a que las precisiones del conjunto de entrenamiento y del conjunto de prueba son muy similares. Una pequeña diferencia indica que el modelo tiene un buen equilibrio y generaliza bien en los datos de prueba, lo que sugiere un buen desempeño sin problemas significativos de sobreajuste o subajuste.

1.7 ¿Cuál es la métrica adecuada para este problema de negocio?

1.7.1 La métrica utilizada en el archivo para evaluar el modelo es la precisión (accuracy). Sin embargo, la métrica adecuada puede depender del contexto específico del negocio y del balance de clases en el conjunto de datos. En muchos problemas de clasificación, además de la precisión, se pueden considerar otras métricas como el F1-score, la precisión, y el recall, especialmente si las clases están desbalanceadas.

1.8 ¿Cuál debería ser el desempeño mínimo a obtener?

1.8.1 Con base a los requisitos de negocio, se proponen el siguiente minimo desempeño:

1.8.2 Accuracy: 0.92

1.8.3 AUC-ROC: 0.90

1.8.4 Recall: 0.70

1.8.5 Precision: 0.95

1.8.6 F1-Score: 0.80

1.8.7 Specificity: 0.95

1.9 El resultado final fue:

1.9.1 El modelo cumple con el desempeño mínimo requerido.