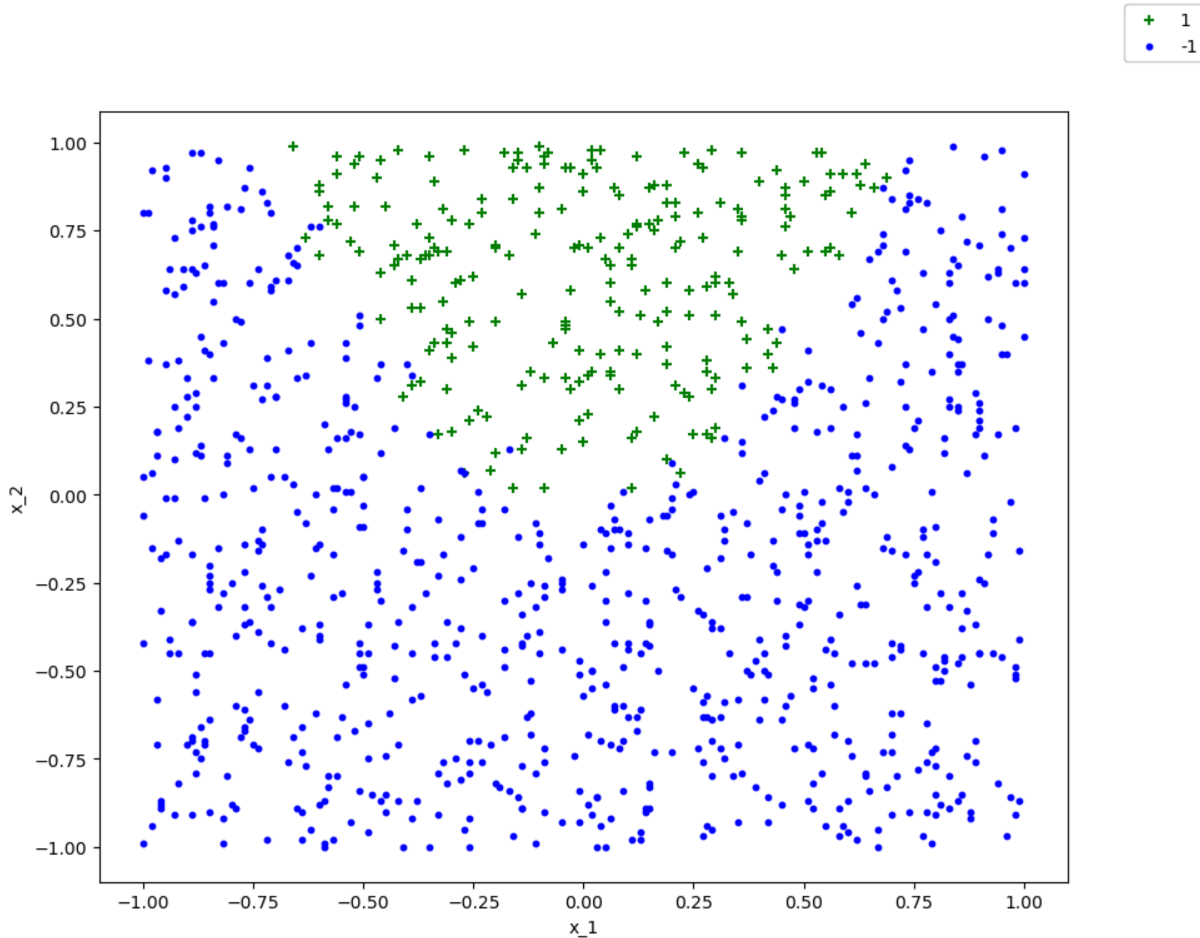
In [7]:
```
import pandas as pd
import numpy as np
# Cargar datos # id:17--34-17
data = pd.read_csv("Introduccion Analitica de datos/dataset1.csv")
data.reset_index(inplace=True)
data.columns = ['X1', 'X2', 'y']
print(data.head())
```

```
     X1    X2  y
0  0.04  0.40  1
1 -0.12 -0.62 -1
2  0.14 -0.42 -1
3 -0.05 -0.93 -1
4  0.60 -0.96 -1
```

In [8]:
```
# Preparar datos
df = data.copy()
X1 = df.iloc[:, 0]
X2 = df.iloc[:, 1]
X = np.column_stack((X1, X2))
y = df.iloc[:, 2]
```

In [28]:
```
# Visualización datos Valores +1 cruz en color verde, valores -1 circulo color azul
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 8))
plt.scatter(X1[y==1], X2[y==1], c='g', marker = '+', label='1')
plt.scatter(X1[y==-1], X2[y==-1], c='b', marker = 'o', label='-1', s=10)
plt.xlabel('x_1')
plt.ylabel('x_2')
plt.legend(bbox_to_anchor=(1.15,1.15), loc='upper right', fancybox=True, framealpha
plt.savefig('Figure_1.png')
plt.show()
```



In [11]:
```
from sklearn.model_selection import train_test_split
# Entrenamiento y prueba aqui afecta que porcentaje de los datos estamos utilizando
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
print('Train Set: ', x_train.shape, y_train.shape)
print('Test Set: ', x_test.shape, y_test.shape)
```

```
Train Set:  (799, 2) (799,)
Test Set:  (200, 2) (200,)
```

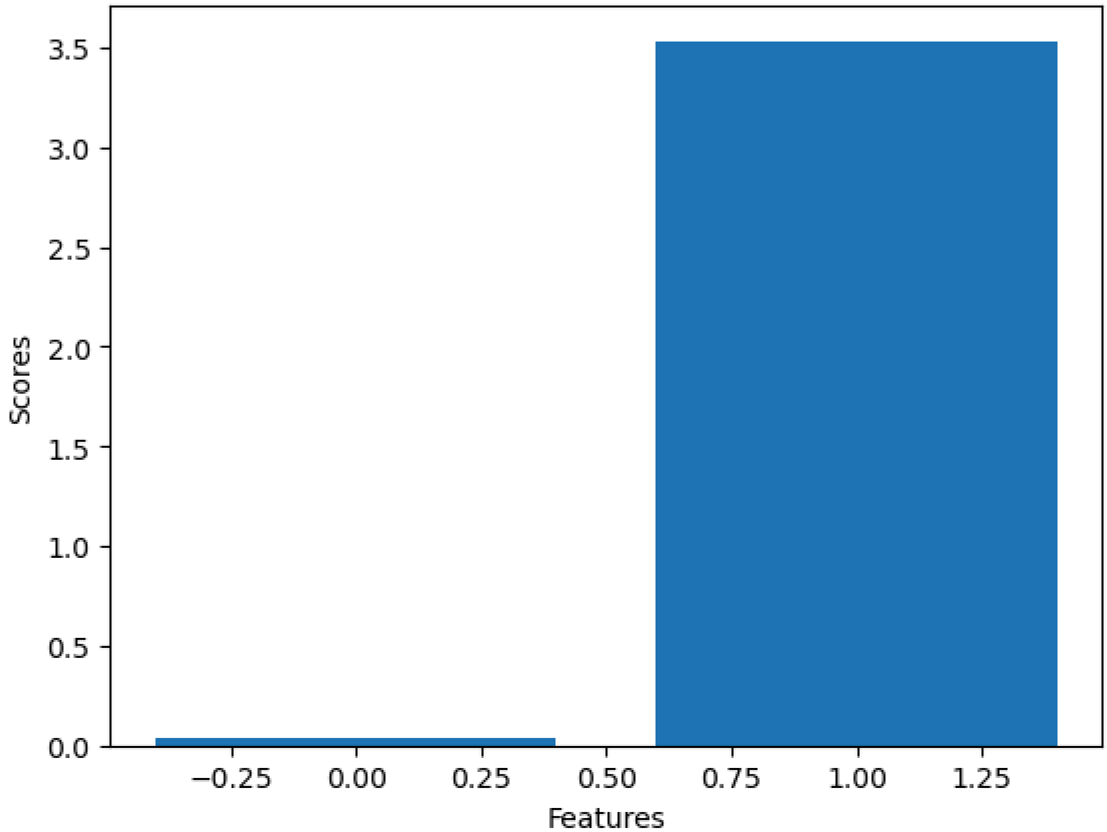
In [17]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# Modelo de Regresión Logística
LR = LogisticRegression()
LR.fit(x_train, y_train)
print('The slopes are: ', LR.coef_[0])
print('The intercept is: ', LR.intercept_)
predictions = LR.predict(x_train)
score = LR.score(x_train, y_train)
print('The score is: ', score)
```

```
The slopes are:  [0.03441392 3.53271057]
The intercept is:  [-2.06876471]
The score is:  0.8197747183979975
```

In [18]:
```
#Feature 1 en este caso tiene un valor absoluto mas grande que feature 0, lo cual i
#En este caso x2 tiene mayor influencia en la prediccion.
feature_importance = LR.coef_[0]
```

```
for i, val in enumerate(feature_importance):
    print('Feature: %0d, Score: %.5f' % (i, val))
plt.bar([x for x in range(len(feature_importance))], feature_importance)
plt.xlabel('Features')
plt.ylabel('Scores')
plt.savefig('Figure_2.png')
```

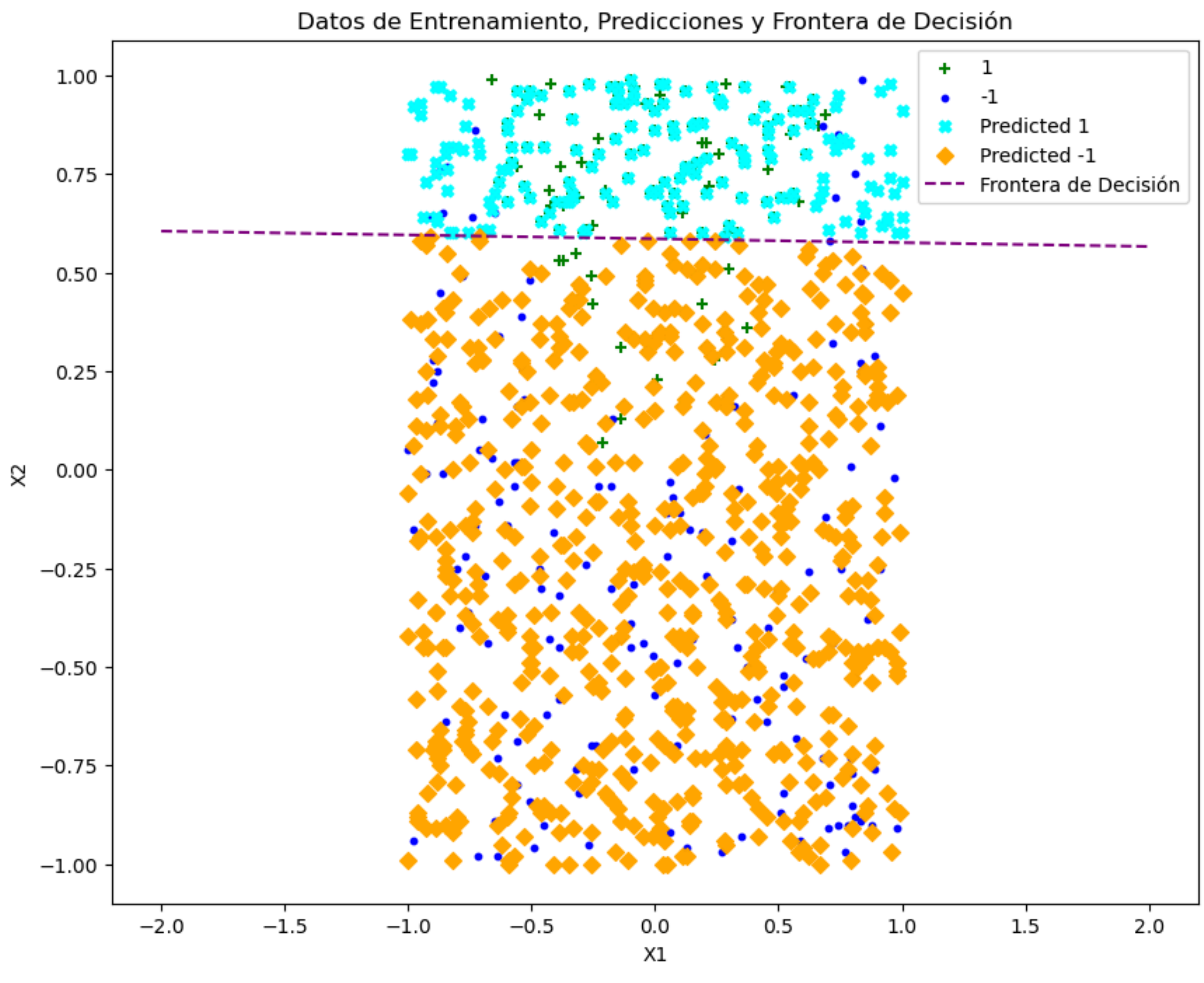
```
Feature: 0, Score: 0.03441
Feature: 1, Score: 3.53271
```



In [36]:
```
from sklearn.metrics import confusion_matrix
from sklearn import metrics
import seaborn as sns

# Crear una figura para el gráfico
plt.figure(figsize=(10, 8))

# Graficar los datos de entrenamiento
plt.scatter(X1[y==1], X2[y==1], c='g', marker='+', label='1')
plt.scatter(X1[y==-1], X2[y==-1], c='b', marker='o', label='-1', s=10)

# Graficar las predicciones
plt.scatter(x_train[predictions == 1][:, 0], x_train[predictions == 1][:, 1], c='cy
plt.scatter(x_train[predictions == -1][:, 0], x_train[predictions == -1][:, 1], c='

# Obtener los coeficientes y el intercepto del modelo
w0 = LR.intercept_[0]
w1, w2 = LR.coef_[0]
print('Intercepto del modelo: ', w0)
print('Pendientes o coeficientes que indican la influencia de las caracteristicas e
```

```python
# Generar valores de X1 para la frontera de decisión
X1_vals = np.linspace(X1.min() - 1, X1.max() + 1, 100)

# Calcular los valores correspondientes de X2 en la frontera de decisión
X2_vals = -(w0 + w1 * X1_vals) / w2

# Dibujar la frontera de decisión
plt.plot(X1_vals, X2_vals, color='purple', linestyle='--', label='Frontera de Decis

# Etiquetas y título del gráfico
plt.xlabel('X1')
plt.ylabel('X2')
plt.title('Datos de Entrenamiento, Predicciones y Frontera de Decisión')
plt.legend()
plt.show()
```

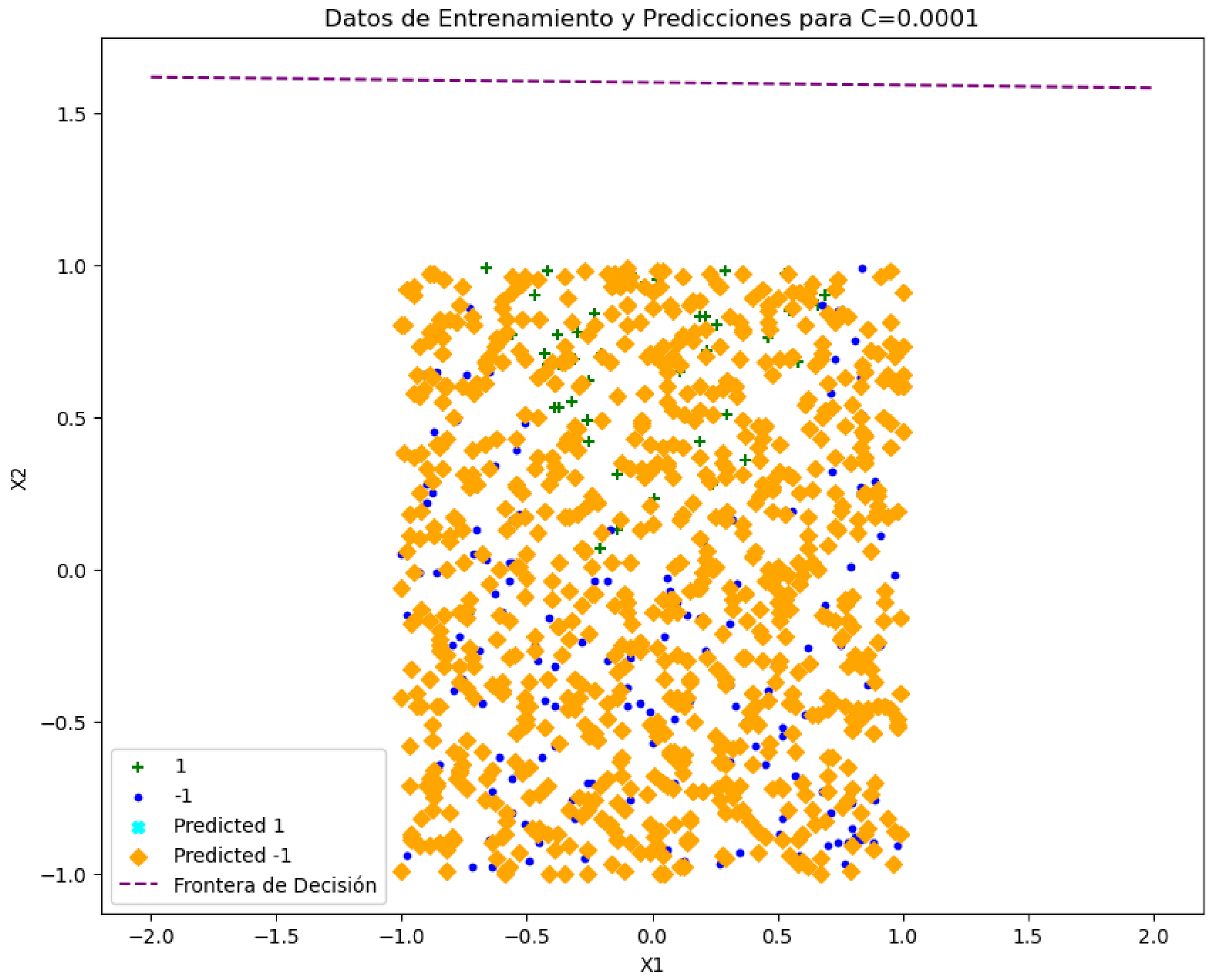
```
Intercepto del modelo:  -2.068764705339932
Pendientes o coeficientes que indican la influencia de las caracteristicas en la pre
dicción:  0.034413917508451394 3.532710568358409
```



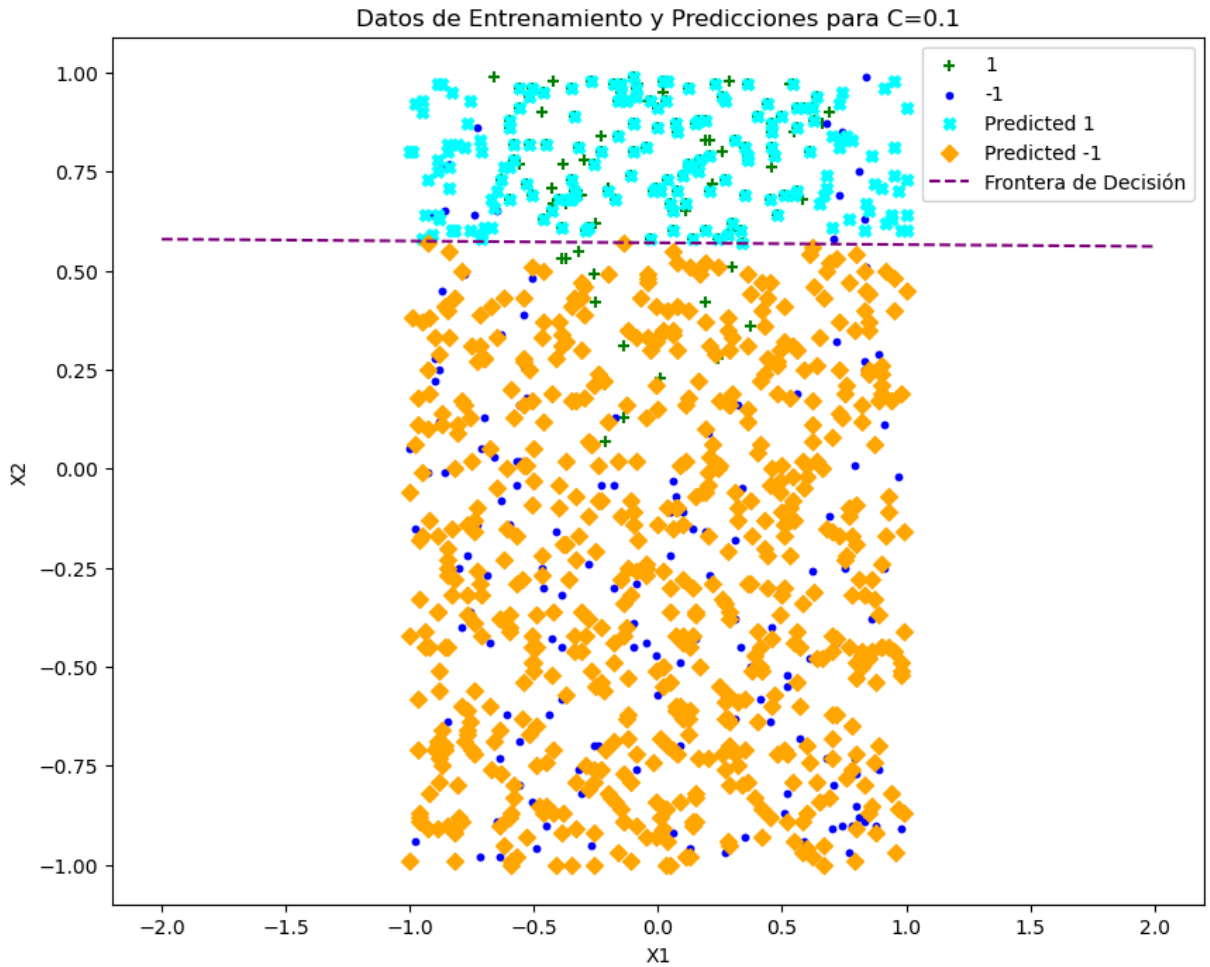
In [88]:
```python
# Parte B Clasificadores SVC lineales
#Lista de valores de C para probar

from sklearn.svm import LinearSVC
from sklearn.metrics import confusion_matrix, classification_report

# Valores de C para probar
```

```
C_values = [0.0001, 0.1, 1, 10, 100, 1000]

# Entrenamiento de modelos SVM lineales
for C in C_values:
    svm_model = LinearSVC(C=C, max_iter=150000)
    svm_model.fit(x_train, y_train)

    print(f'Modelo SVM con C={C}:')
    print('Coeficientes:', svm_model.coef_)
    print('Intercepto:', svm_model.intercept_)

    # Predicciones
    predictions = svm_model.predict(x_train)
    svm_score = svm_model.score(x_train, y_train)
    print(f"Precisión: {svm_score}")

    # Reporte de clasificación
    report = classification_report(y_train, predictions, zero_division=1)
    print(f'Reporte de Clasificación para C={C}:\n{report}')

    # Visualización de predicciones
    plt.figure(figsize=(10, 8))
    plt.scatter(X1[y == 1], X2[y == 1], c='g', marker='+', label='1')
    plt.scatter(X1[y == -1], X2[y == -1], c='b', marker='o', label='-1', s=10)
    plt.scatter(x_train[predictions == 1][:, 0], x_train[predictions == 1][:, 1], c
    plt.scatter(x_train[predictions == -1][:, 0], x_train[predictions == -1][:, 1],

    # Frontera de decisión
    w0 = svm_model.intercept_[0]
    w1, w2 = svm_model.coef_[0]
    X1_vals = np.linspace(X1.min() - 1, X1.max() + 1, 100)
    X2_vals = -(w0 + w1 * X1_vals) / w2
    plt.plot(X1_vals, X2_vals, color='purple', linestyle='--', label='Frontera de D

    # Etiquetas y título
    plt.xlabel('X1')
    plt.ylabel('X2')
    plt.title(f'Datos de Entrenamiento y Predicciones para C={C}')
    plt.legend()
    plt.show()
```

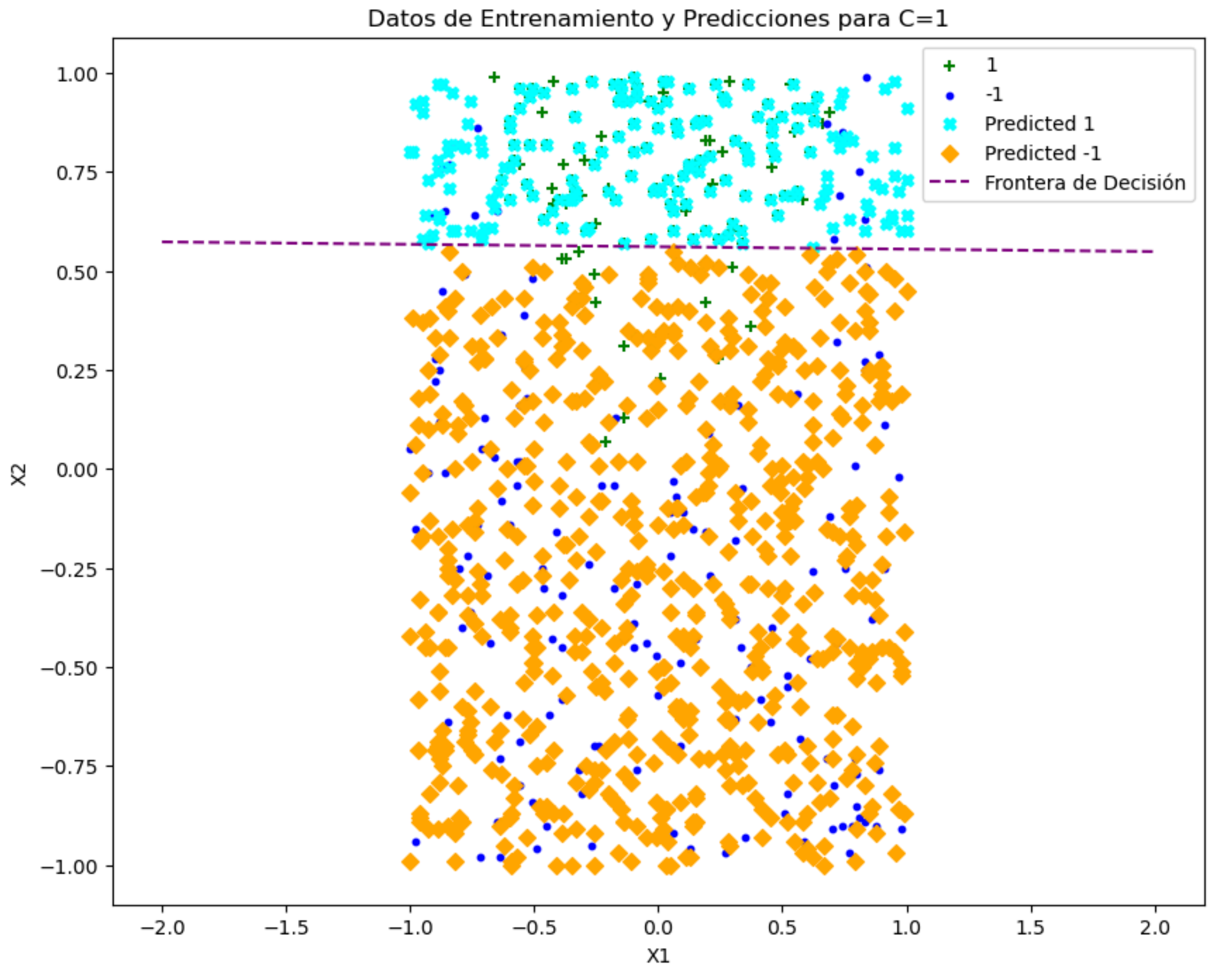
```
Modelo SVM con C=0.0001:
Coeficientes: [[0.00040128 0.04556092]]
Intercepto: [-0.07284952]
Precisión: 0.7647058823529411
Reporte de Clasificación para C=0.0001:
              precision    recall  f1-score   support

          -1       0.76      1.00      0.87       611
           1       1.00      0.00      0.00       188

    accuracy                           0.76       799
   macro avg       0.88      0.50      0.43       799
weighted avg       0.82      0.76      0.66       799
```



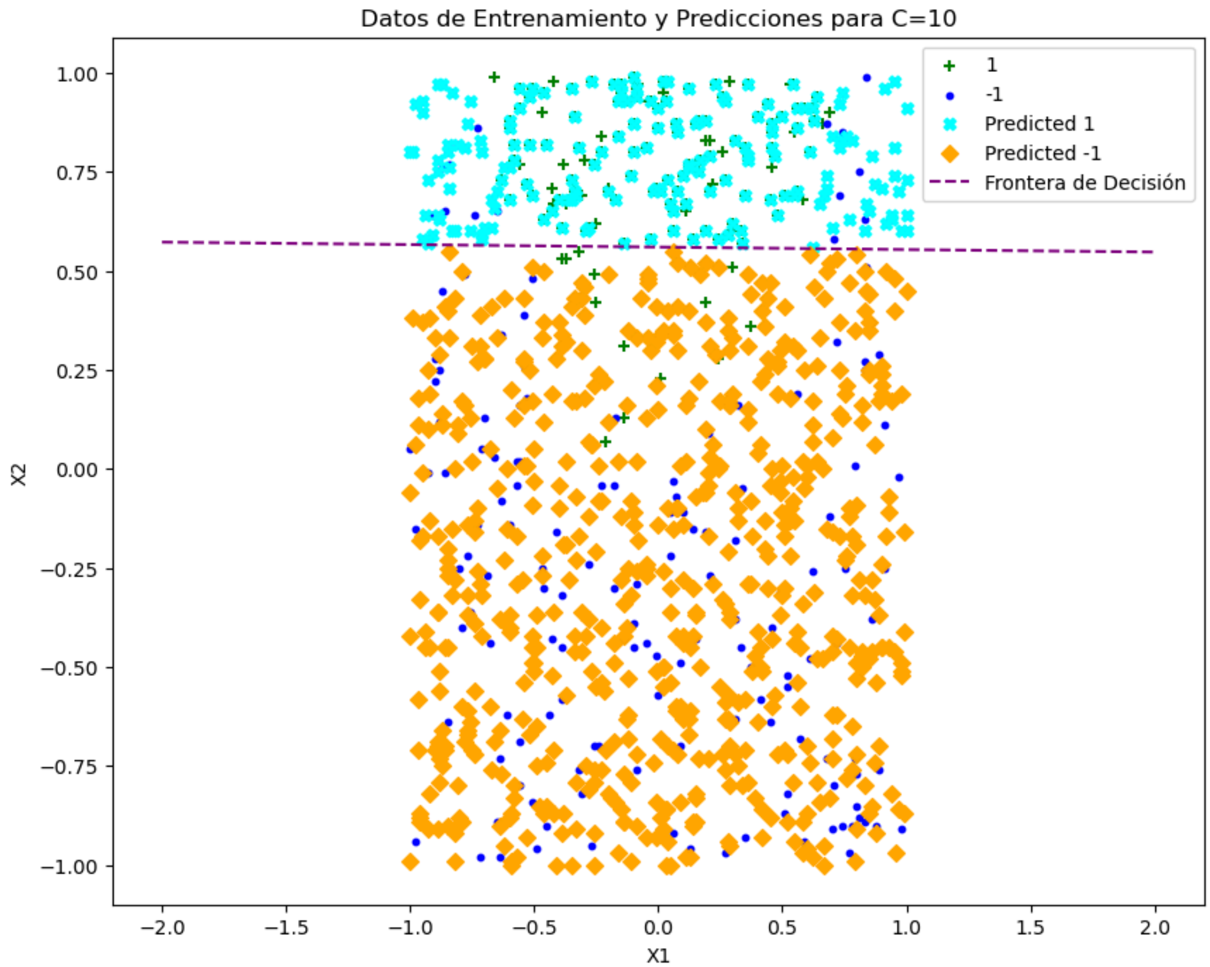

```
Modelo SVM con C=0.1:
Coeficientes: [[0.00543859 1.1886607 ]]
Intercepto: [-0.67780149]
Precisión: 0.8197747183979975
Reporte de Clasificación para C=0.1:
              precision    recall  f1-score   support

          -1       0.88      0.89      0.88       611
           1       0.62      0.61      0.61       188

    accuracy                           0.82       799
   macro avg       0.75      0.75      0.75       799
weighted avg       0.82      0.82      0.82       799
```



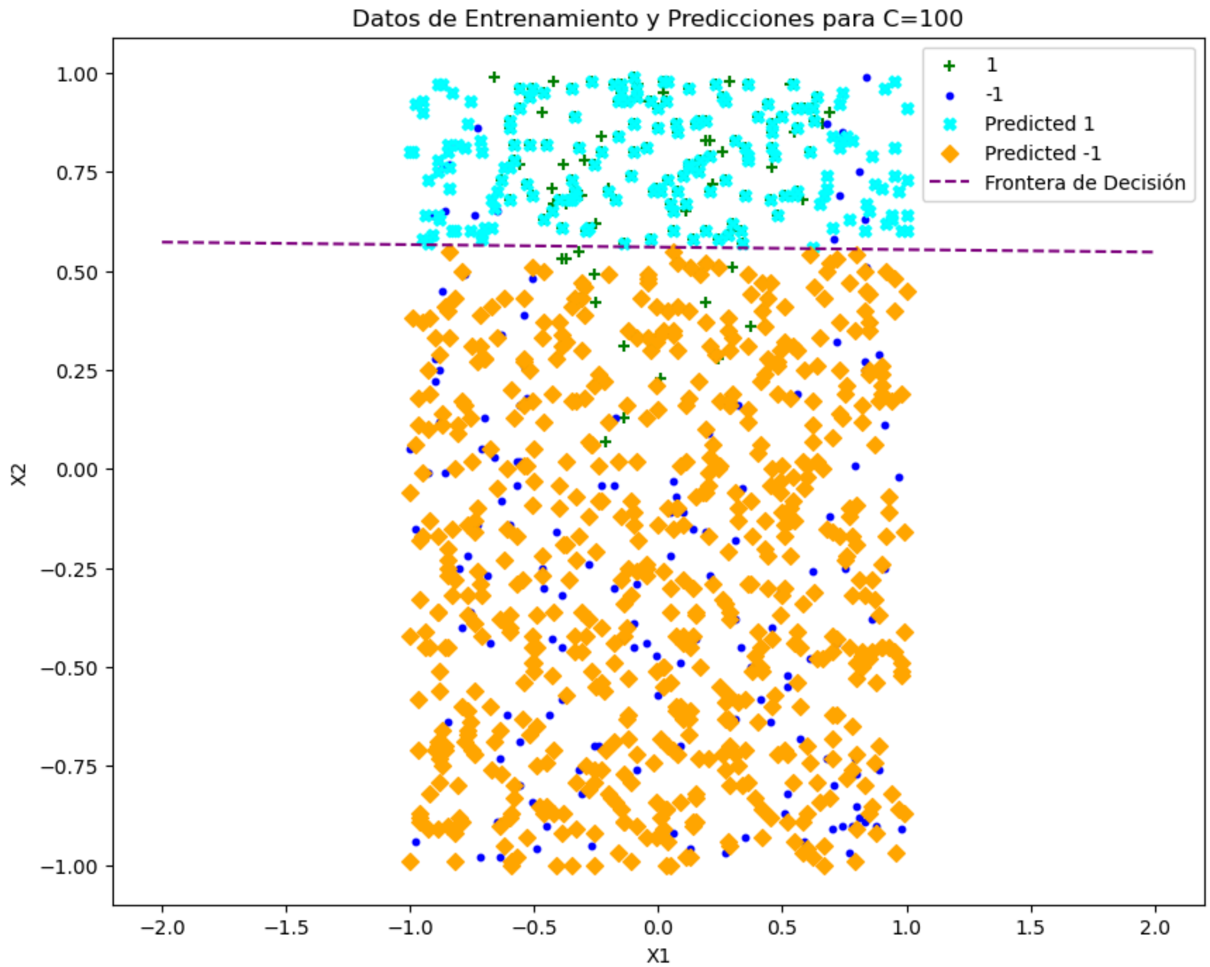

```
Modelo SVM con C=1:
Coeficientes: [[0.00789962 1.29363758]]
Intercepto: [-0.72567207]
Precisión: 0.818523153942428
Reporte de Clasificación para C=1:
              precision    recall  f1-score   support

          -1       0.88      0.88      0.88       611
           1       0.61      0.61      0.61       188

    accuracy                           0.82       799
   macro avg       0.75      0.75      0.75       799
weighted avg       0.82      0.82      0.82       799
```



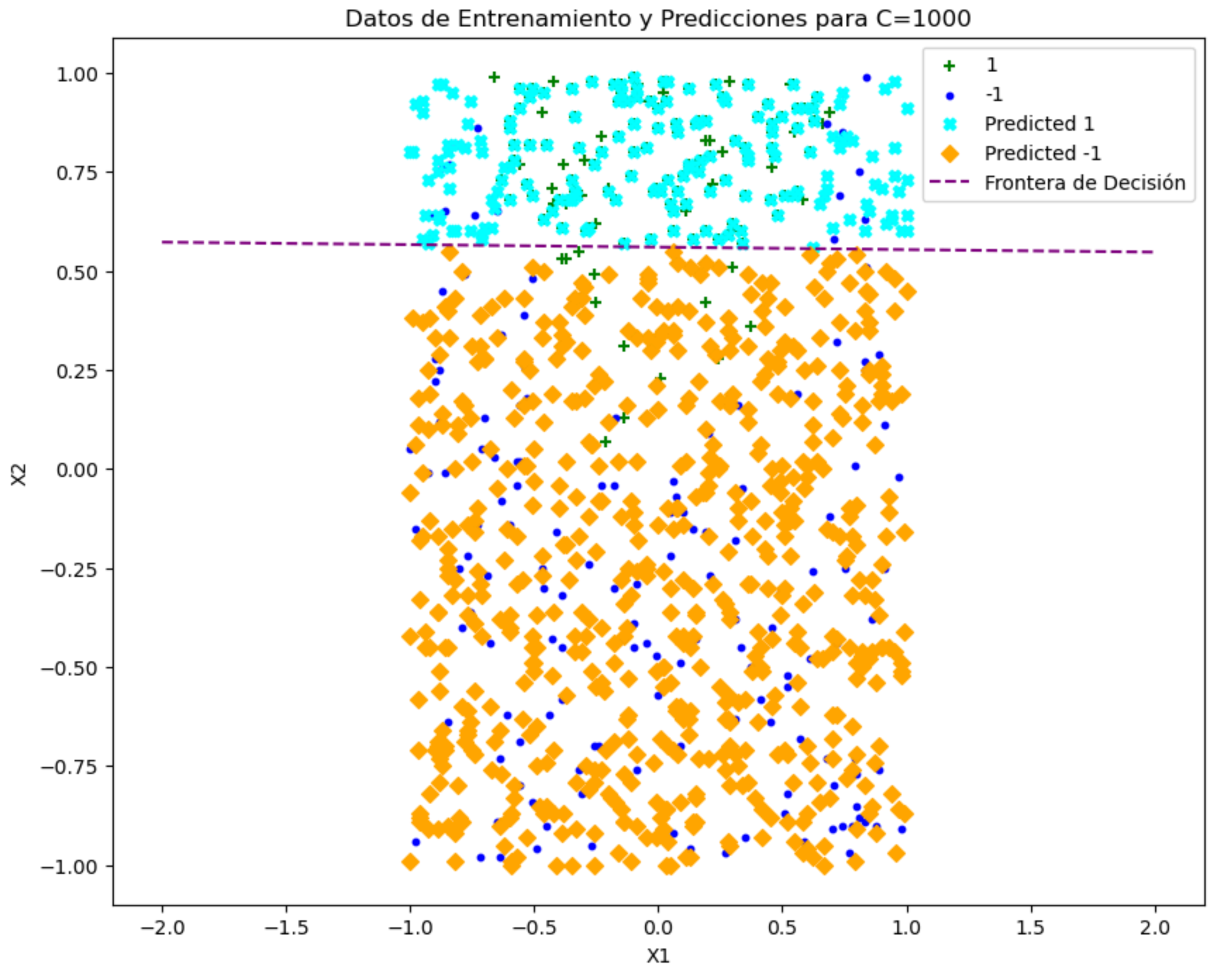

```
Modelo SVM con C=10:
Coeficientes: [[0.00817198 1.30598343]]
Intercepto: [-0.73141723]
Precisión: 0.818523153942428
Reporte de Clasificación para C=10:
              precision    recall  f1-score   support

          -1       0.88      0.88      0.88       611
           1       0.61      0.61      0.61       188

    accuracy                           0.82       799
   macro avg       0.75      0.75      0.75       799
weighted avg       0.82      0.82      0.82       799
```




```
Modelo SVM con C=100:
Coeficientes: [[0.00820031 1.30724005]]
Intercepto: [-0.73200435]
Precisión: 0.818523153942428
Reporte de Clasificación para C=100:
              precision    recall  f1-score   support

          -1       0.88      0.88      0.88       611
           1       0.61      0.61      0.61       188

    accuracy                           0.82       799
   macro avg       0.75      0.75      0.75       799
weighted avg       0.82      0.82      0.82       799
```




```
Modelo SVM con C=1000:
Coeficientes: [[0.00820414 1.30737412]]
Intercepto: [-0.73206439]
Precisión: 0.818523153942428
Reporte de Clasificación para C=1000:
              precision    recall  f1-score   support

          -1       0.88      0.88      0.88       611
           1       0.61      0.61      0.61       188

    accuracy                           0.82       799
   macro avg       0.75      0.75      0.75       799
weighted avg       0.82      0.82      0.82       799
```


Datos de Entrenamiento y Predicciones para C=1000
1
-1
Predicted 1
Predicted -1
Frontera de Decisión
X2
X1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

In [ ]: