Nombre:

Samano Cardenas Andrea Guadalupe

Materia:

Inteligencia Artificial

Carrera:

Ing. Sistemas Computacionales

TAREA 5:

Clasificación con SVM con función radial

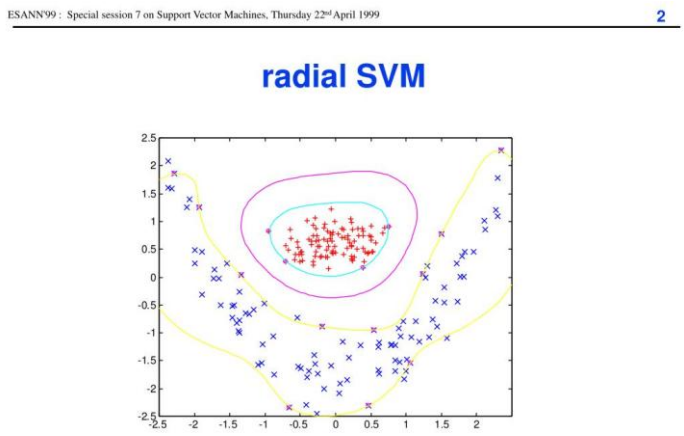
Clasificación con SVM con función radial

¿Qué es un SVM?

Las Máquinas de Vectores de Soporte (SVM) son un algoritmo de aprendizaje aut omático supervisado que se utiliza principalmente para problemas de clasificación. El objetivo de SVM es encontrar un hiperplano que separe las diferentes clases d e datos con el mayor margen posible. Este margen se define como la distancia ent re el hiperplano y los puntos de datos más cercanos de cada clase, conocidos com o vectores de soporte.

El kernel RBF es una de las funciones de kernel más populares utilizadas en SVM. Permite que el algoritmo maneje datos que no son linealmente separables al trans formar el espacio de entrada en un espacio de dimensión infinita. Esto significa qu e SVM puede encontrar un hiperplano que separe las clases de manera efectiva, i ncluso en situaciones complejas donde los datos están distribuidos de forma no lin eal.

La función kernel de base radial (RBF) es una de las funciones de kernel más utiliz adas en SVM. Se utiliza cuando los datos no lineales se pueden separar utilizando un hiperplano no lineal. La función kernel RBF se define como: donde gamma es un parámetro que controla la amplitud de la función kernel. Un valor más alto de g amma se ajustará perfectamente al conjunto de datos de entrenamiento, lo que pr ovoca un sobreajuste. Gamma = 0,1 se considera un buen valor por defecto.



TIPOS DE CLASIFICADORES DE SVM

SVM lineales

Las SVM lineales se emplean con datos separables linealmente. Esto significa que los datos no necesitan someterse a ninguna transformación para separar los datos en diferentes clases. El límite de decisión y los vectores de soporte forman la apariencia de una calle, y el profesor Patrick Winston del MIT emplea la analogía de "ajustar la calle más ancha posible"[2] (enlace externo a ibm.com) para describir este problema de optimización cuadrática. Matemáticamente, este hiperplano de separación se puede representar de la siguiente manera: $wx + b = 0$

SVM no lineales
Gran parte de los datos del mundo real no son linealmente separables, y ahí es donde entran en juego las SVM no lineales. Para que los datos sean linealmente separables, se aplican métodos de preprocesamiento a los datos de entrenamiento para transformarlos en un espacio de características de mayor dimensión. Dicho esto, los espacios de mayor dimensión pueden crear más complejidad al aumentar el riesgo de sobreajuste de los datos y convertirse en una carga computacional. El "truco del núcleo" ayuda a reducir parte de esa complejidad, haciendo que el cálculo sea más eficiente, y lo hace sustituyendo los cálculos del producto de punto por una función de núcleo equivalente.

Creación de un clasificador SVM

- **Dividir sus datos**
  Al igual que con otros modelos de aprendizaje automático, comience dividiendo sus datos en un conjunto de entrenamiento y un conjunto de prueba. Por otro lado, esto supone que ya realizó un análisis exploratorio de sus datos. Si bien esto no es técnicamente necesario para crear un clasificador SVM, es una buena práctica antes de usar cualquier modelo de aprendizaje automático, ya que esto le permitirá conocer los datos faltantes o los valores atípicos.

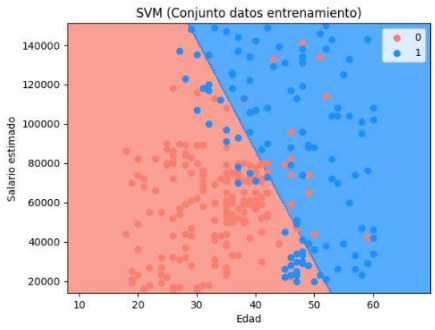
- **Generar y evaluar el modelo**
  Importe un módulo SVM de la biblioteca de su elección, como scikit-learn (enlace externo a ibm.com). Entrene a sus muestras de entrenamiento en el clasificador y prediga la respuesta. Puede evaluar el rendimiento comparando la precisión del conjunto de prueba con los valores previstos. Es recomendable que emplee otras métricas de evaluación, como el valor f1, la precisión o la recuperación.

- **Ajuste de hiperparámetros**
  Los hiperparámetros se pueden ajustar para mejorar el rendimiento de un modelo SVM. Los hiperparámetros óptimos se pueden encontrar utilizando métodos de búsqueda de cuadrícula y validación cruzada, que iterarán a través de diferentes valores de kernel, regularización (C) y gamma para encontrar la mejor combinación.

EJEMPLOS DE CODIGOS

Python

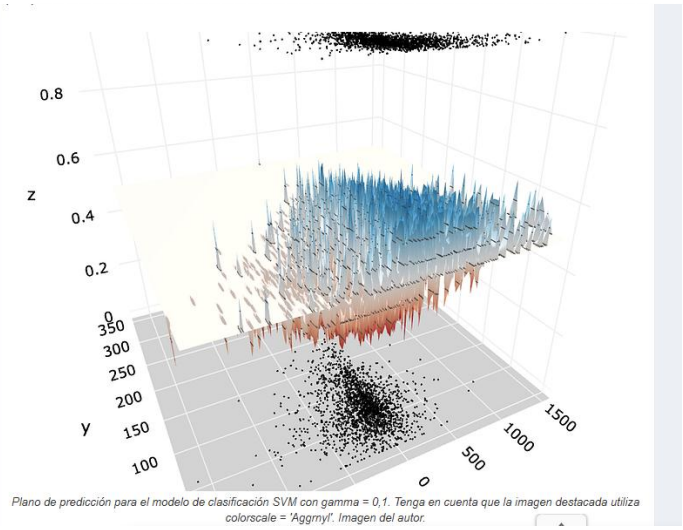
```
# Importar las bibliotecas necesarias
import numpy as np
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score

# Cargar un conjunto de datos (por ejemplo, Iris)
data = datasets.load_iris()
X = data.data  # Características
y = data.target  # Etiquetas

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Crear el modelo SVM con kernel radial (RBF)
svm_rbf = SVC(kernel='rbf', C=1.0, gamma='scale')  # Puedes ajustar C y gamma
según sea necesario

# Entrenar el modelo
svm_rbf.fit(X_train, y_train)

# Realizar predicciones
y_pred = svm_rbf.predict(X_test)

# Evaluar el modelo
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nReporte de Clasificación:\n", classification_report(y_test, y_pred))
```



Modelo SVM 2 - Gamma = 0.1

```
# Select data for modeling
X=df[['rating_difference', 'turns']]
y=df['white_win'].values

# Fit the model and display results
X_train, X_test, y_train, y_test, clf = fitting(X, y, 1, 0.1)

# Plot 3D chart
Plot_3D(X, X_test, y_test, clf)
```

```
----- Evaluation on Test Data -----
Accuracy Score:  0.603938185443669
              precision    recall  f1-score   support

           0       0.60      0.64      0.62      2024
           1       0.61      0.57      0.59      1988

    accuracy                           0.60      4012
   macro avg       0.60      0.60      0.60      4012
weighted avg       0.60      0.60      0.60      4012

------------------------------------------------------------
----- Evaluation on Training Data -----
Accuracy Score:  0.8003240683036271
              precision    recall  f1-score   support

           0       0.80      0.81      0.80      8033
           1       0.80      0.80      0.80      8013

    accuracy                           0.80     16046
   macro avg       0.80      0.80      0.80     16046
weighted avg       0.80      0.80      0.80     16046
```



Plano de predicción para el modelo de clasificación SVM con gamma = 0,1. Tenga en cuenta que la imagen destacada utiliza colorscale = 'Aggrnyl'. Imagen del autor.

Modelo SVM 3: Gamma = 0,000001

```
# Select data for modeling
X=df[['rating_difference', 'turns']]
y=df['white_win'].values

# Fit the model and display results
X_train, X_test, y_train, y_test, clf = fitting(X, y, 1, 0.000001)

# Plot 3D chart
Plot_3D(X, X_test, y_test, clf)
```

```
----- Evaluation on Test Data -----
Accuracy Score:  0.6602691924227319
              precision    recall  f1-score   support

           0       0.65      0.70      0.68      2024
           1       0.67      0.62      0.64      1988

    accuracy                           0.66      4012
   macro avg       0.66      0.66      0.66      4012
weighted avg       0.66      0.66      0.66      4012

------------------------------------------------------------
----- Evaluation on Training Data -----
Accuracy Score:  0.6463916240807678
              precision    recall  f1-score   support

           0       0.64      0.67      0.65      8033
           1       0.65      0.62      0.64      8013

    accuracy                           0.65     16046
   macro avg       0.65      0.65      0.65     16046
weighted avg       0.65      0.65      0.65     16046
```