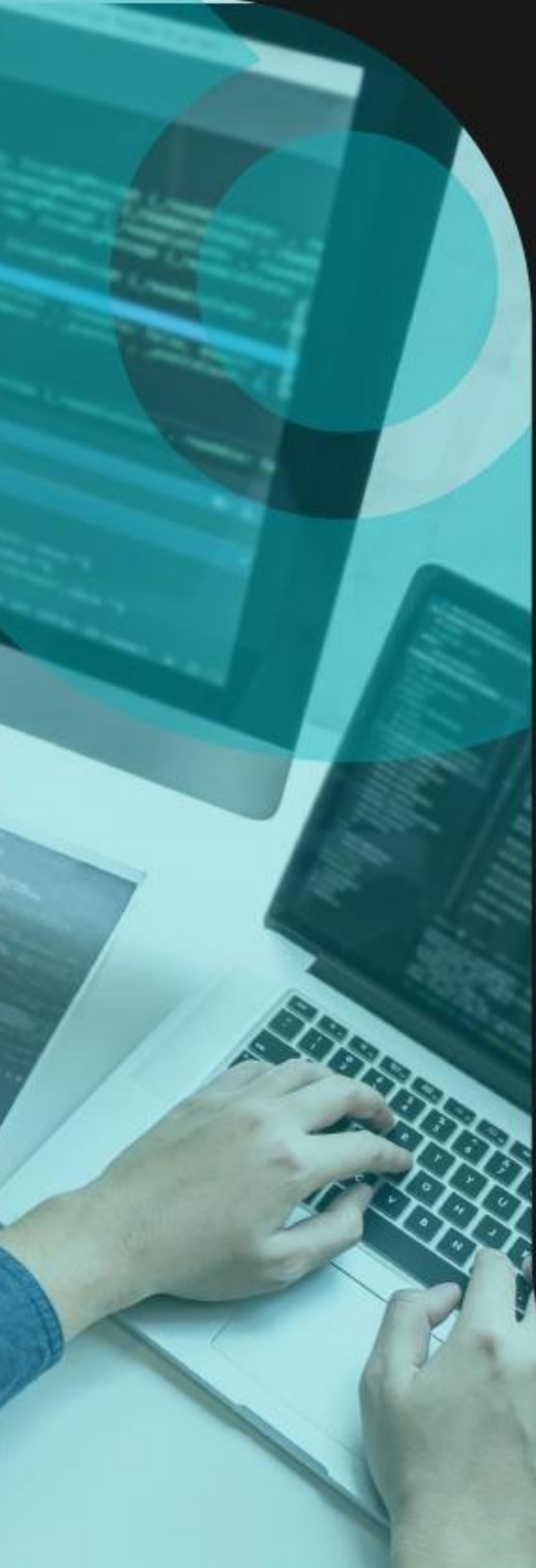


AWAKELAB

BASECAMP

Ciencia de Datos

Módulo: Aprendizaje de Máquina Supervisado

Aprendizaje Esperado

---

8. Elaborar un modelo predictivo aplicando el algoritmo clasificador SVM para resolver un problema de clasificación utilizando lenguaje Python.

Máquinas de Soporte Vectorial

¿Qué es?

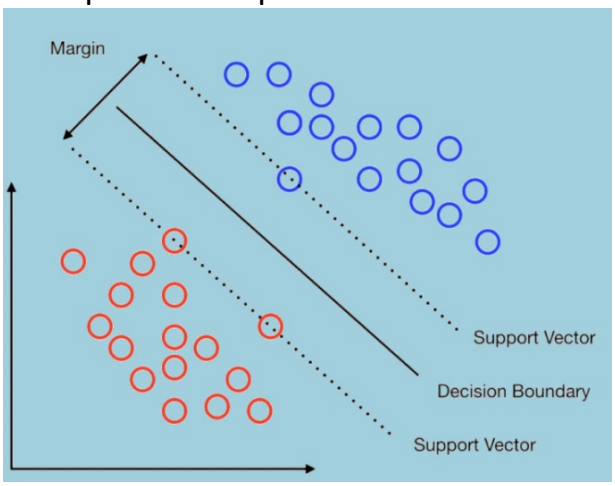
Conocido también como Support Vector Machine y corresponde a una técnica de Machine learning de tipo Supervisado. Son una de las técnicas de clasificación y predicción más precisas entre los modelos no jerárquicos. También puede utilizarse para resolver un problema de regresión.

Una de las claves de su buen rendimiento es que busca encontrar el hiperplano que mejor separa dos categorías, maximizando el margen de separación entre ellas. La idea es determinar un hiperplano que clasifique correctamente los registros de cada clase.

Cómo es el clasificador

- El mejor hiperplano para clasificar las 2 clases es aquél que maximiza el margen de separación entre ellas.
- El margen de separación se define como la distancia perpendicular entre el hiperplano (*decision boundary*) y los registros más cercanos a cada uno de sus lados.
- Es importante notar que la posición de este hiperplano es definida por un reducido conjunto de los datos de entrenamiento. Estos

registros o vectores son los que dan soporte a la frontera de decisión, de ahí el nombre del método.



SVM - Caso lineal y separable

Determinar este hiperplano implica 2 condiciones fundamentales:

1. Cada registro del set de entrenamiento debe ser bien clasificado, equivalente a decir que cada punto está del lado correcto del hiperplano. Esto asume que los datos son *linealmente separables*.
2. El hiperplano debe ser aquél que maximiza el margen deseado (*margin*).

Problema de Optimización
Sea $z_k$variable auxiliar que toma el valor 1 o -1 dependiendo de si la observación x_k pertenece a C_1 *o* C_2.

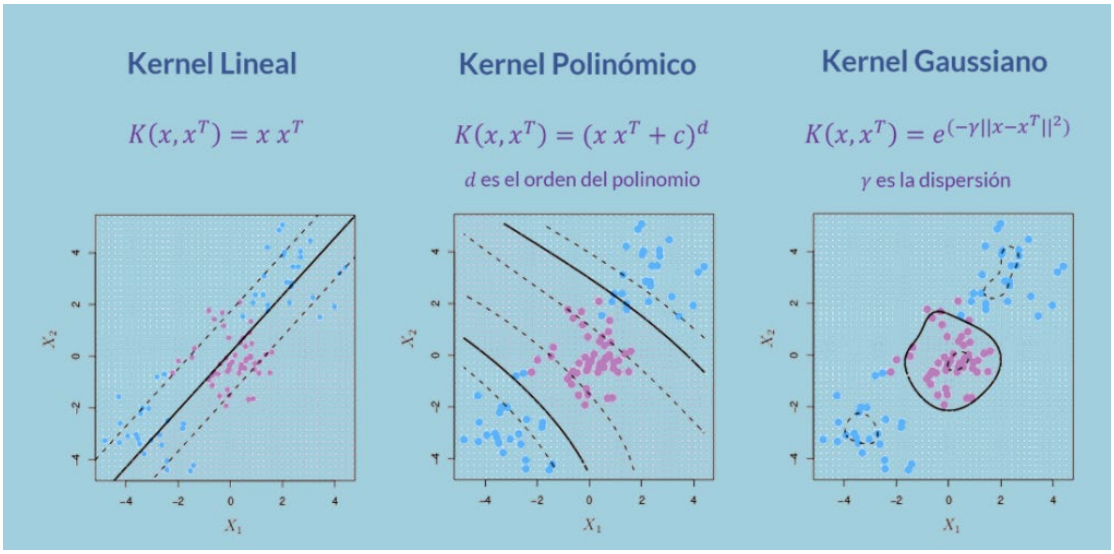
Las dos condiciones anteriores se traducen en el siguiente problema de optimización:

$$argmax_{\omega,c}\left\{\frac{1}{||\omega||}z_k g(x_k)\right\} \quad (condición\ 2)$$

Sujeto a $z_k(\omega x_k + c) > 0, \forall k \in 1, \dots, n \quad (condición\ 1)$.

SVM - Caso no lineal pero separable

Los kernels son funciones que permiten convertir lo que sería un problema de clasificación no lineal en el espacio dimensional original, a un sencillo problema de clasificación lineal en un espacio dimensional mayor.



Tipos de Kernel SVM

- **Kernel lineal**

Es el tipo más básico de kernel, generalmente de naturaleza unidimensional. Demuestra ser la mejor función cuando hay muchas características. El núcleo lineal se prefiere principalmente para problemas de clasificación de texto , ya que la mayoría de estos tipos de problemas de clasificación se pueden separar linealmente.

Las funciones de núcleo lineal son más rápidas que otras funciones.

Fórmula de kernel lineal
F(x, xj) = suma(x.xj)

Aquí, x, xj representa los datos que intenta clasificar.

- **Kernel polinomial**

Es una representación más generalizada del núcleo lineal. No es tan preferida como otras funciones del kernel, ya que es menos eficiente y precisa.

- **Función de base radial gaussiana (RBF)**

Es una de las funciones del núcleo más preferidas y utilizadas en svm. Por lo general, se elige para datos no lineales. Ayuda a hacer una separación adecuada cuando no hay un conocimiento previo de los datos.

El valor de gamma varía de 0 a 1 . Debe proporcionar manualmente el valor de gamma en el código. El valor más preferido para gamma es 0,1 .

- **Kernel sigmoide**

Se prefiere sobre todo para las redes neuronales . Esta función del núcleo es similar a un modelo de perceptrón de dos capas de la red neuronal, que funciona como una función de activación para las neuronas.

- **Kernel gaussiano**

Es un núcleo de uso común. Se utiliza cuando no hay conocimiento previo de un conjunto de datos determinado.

- **Kernel ANOVA**

También se conoce como kernel de función de base radial. Suele funcionar bien en problemas de regresión multidimensional.

Ventajas	**Desventajas**

Ofrecen buena precisión y realizan predicciones más rápidas en comparación con el algoritmo de Naive Bayes.	No son adecuadas para grandes conjuntos de datos debido a su alto tiempo de formación y también requiere más tiempo de formación en comparación con Naive Bayes.
Utilizan menos memoria porque utilizan un subconjunto de puntos de entrenamiento en la fase de decisión.	Funciona mal con clases superpuestas y también es sensible al tipo de núcleo utilizado.
Este algoritmo funciona bien con un claro margen de separación y con un espacio dimensional elevado.	

Ejemplo en Python

La base de datos breast cancer contiene distintas características de pacientes con cáncer de mama, la variable target consiste en si el tumor es benigno o maligno. La idea es lograr separar estas dos clases mediante un hiperplano usando un modelo de support vector machine.

```
from sklearn import datasets

cancer = datasets.load_breast_cancer()

from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test =
train_test_split(cancer.data, cancer.target,
test_size=0.3,random_state=0) #split

from sklearn import svm

svm_linear = svm.SVC(kernel='linear') #SVM con kernel
lineal

svm_linear.fit(X_train, y_train)

y_predicted = svm_linear.predict(X_test) #predicción

from sklearn import metrics #métricas

print("Accuracy:",metrics.accuracy_score(y_test,
y_predicted))

print("Precision:",metrics.precision_score(y_test,
y_predicted))

print("Recall:",metrics.recall_score(y_test,
y_predicted)) #Sensibilidad

# Accuracy: 0.9590643274853801
      # Precision: 0.9809523809523809
      # Recall: 0.9537037037037037
```

Referencias

[1] Qué es SVM

https://www.cienciadedatos.net/documentos/34_maquinas_de_vector_soporte_support_vector_machines

[2] SVM

https://scikit-learn.org/stable/modules/svm.html

Material Complementario

[1] Cómo interpretar SVM

https://www.youtube.com/watch?v=QoRBenaGzzw

[2] Cómo funciona SVM

https://www.youtube.com/watch?v=kl6tyEi5eso