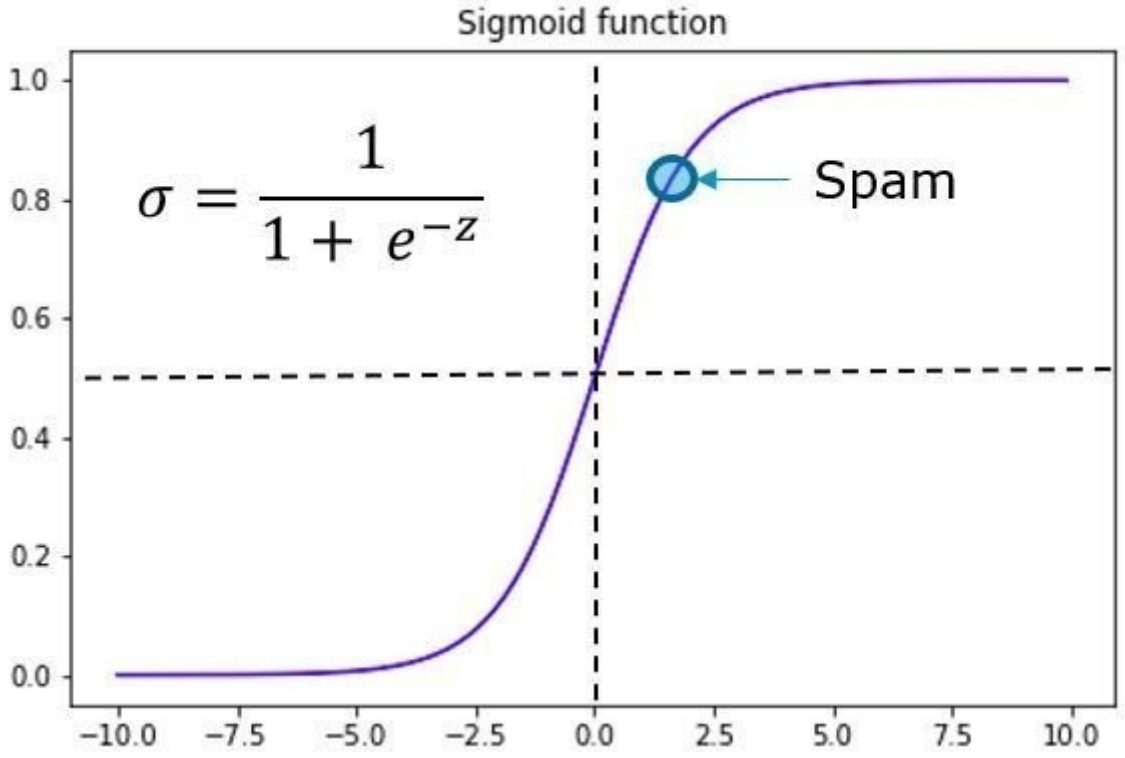

¿Cómo usar Regresión Logística en Python?

La regresión logística es una técnica de aprendizaje supervisado para clasificación. Es muy usada en muchas industrias debido a su escalabilidad y explicabilidad.

Instrucciones rápidas

¿Cómo usar Regresión Logística en Python con scikit-learn?

Importa la librería numérica NumPy

```
import numpy as np
```

Prepara los datos de entrenamiento

X serán los datos de entrada, y los de salida en este ejemplo

Importa el módulo LogisticRegression de la librería scikit-learn

```
from sklearn.linear_model import LogisticRegression
```

Crea una instancia de la Regresión Logística

```
regresion_logistica = LogisticRegression()
```

Entrena la regresión logística con los datos de entrenamiento

regresion_logistica.fit(X,y)

Usa el modelo entrenado para obtener las predicciones con datos nuevos

prediccion = regresion_logistica.predict(X_nuevo)

Opcionalmente, obtén las probabilidades de la predicción

probabilidades_prediccion = regresion_logistica.predict_proba(X_nuevo)

Ejemplo de Regresión Logística en Python

Datos

Vamos a suponer que queremos predecir cuál es la probabilidad que tiene un estudiante de aprobar un examen en función de las horas que ha estudiado. Date cuenta que para 1.75 horas de estudio, hay un estudiante que aprueba y el otro que no.

Horas	0.50	0.75	1.00	1.25	1.50	1.75	1.75	2.00	2.25	2.50
Aprueba	0	0	0	0	0	0	1	0	1	0

Horas	2.75	3.00	3.25	3.50	4.00	4.25	4.50	4.75	5.00	5.50
Aprueba	1	0	1	0	1	1	1	1	1	1

Podemos escribir el siguiente código python para representar estos datos:

```
# Paso 1: importamos la librería numérica NumPy

import numpy as np
```

```
# Paso 2: preparamos los datos

X = np.array([0.5, 0.75, 1, 1.25, 1.5, 1.75, 1.75, 2, 2.25, 2.5, 2.75, 3, 3.25, 3.5, 4, 4.25, 4.5, 4.75, 5, 5.5]).reshape(-1,1)

y = np.array([0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1])
```

Entrenando la regresión logística

Durante la fase de entrenamiento, el modelo aprende qué coeficientes minimizan la función de coste.

```
# Paso 3: importamos la clase LogisticRegresion de scikit-learn

from sklearn.linear_model import LogisticRegression

# Paso 4: Creamos una instancia de la Regresión Logística

regresion_logistica = LogisticRegression()

# Paso 5: Entrena la regresión logística con los datos de entrenamiento

regresion_logistica.fit(X,y)
```

Haciendo predicciones

Vamos a ver cómo podemos hacer predicciones una vez que el modelo está entrenado. Primero haremos predicciones absolutas y luego predicciones relativas. Vamos a ver qué pasa si estudiamos 1, 2, 3, 4, 5 ó 6 horas.

```
X_nuevo = np.array([1, 2, 3, 4, 5, 6]).reshape(-1,1)

# Paso 6: Usa el modelo entrenado para obtener las predicciones con datos nuevos

prediccion = regresion_logistica.predict(X_nuevo)
```

```
print(prediccion)
```

```
# produce el resultado: [0 0 1 1 1 1]
```

Como podemos ver, en el caso que el estudiemos 1 ó 2 horas, lo más probable es que suspendamos para este examen. Si estudiamos 3 o más horas, lo más probable es aprobar. Vamos a ver ahora cómo podemos calcular las probabilidades en estos casos.

```
# Paso 7: Opcionalmente, obtén las probabilidades de la predicción

probabilidades_prediccion =
regresion_logistica.predict_proba(X_nuevo)
print(probabilidades_prediccion)


# produce el siguiente resultado (la primera columna es
# la probabilidad de suspender y la segunda columna es
# la probabilidad de aprobar)
# [[0.6801015  0.3198985 ]
#  [0.53568295 0.46431705]
#  [0.38502138 0.61497862]
#  [0.25359079 0.74640921]
#  [0.15566862 0.84433138]
#  [0.09095092 0.90904908]]

# Como seguramente estamos más interesados en la probabilidad de
aprobar,
# podemos centrarnos en la segunda columna

print(probabilidades_prediccion[:,1])



# produce el resultado:

[0.3198985  0.46431705 0.61497862 0.74640921 0.84433138
0.90904908]
```