



UNIVERSIDAD SANTO TOMÁS

PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA

SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732



Acreditación Institucional
Internacional

OTORGADA POR EL IAC CINDE ACUERDO 55 DEL 8 DE MAYO-VIGENCIA 5 AÑOS



ACREDITACIÓN
INSTITUCIONAL
DE ALTA CALIDAD
MULTICAMPUS

Res. MEN No. 03495 del 29 de enero de 2010
Vigencia por seis años



QS STARS
RATED FOR EXCELLENCE



ISO 9001
Acreditación

ISO 9001-1



CERTIFIED
IONet
MANAGEMENT SYSTEM



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732

Faculty: Systems engineer
Course: Deep Learning
Topic: Sklearn – regresión logística

Professor: Luis Fernando Castellanos Guarín
Email: Luis.castellanosg@usantoto.edu.co
Phone: 3214582098



Clasificadores (algoritmos) de aprendizaje automático con SCIKIT-LEARN

**¡Siempre
hacia lo alto!**



CONTENIDO

- Los Algoritmos más populares para clasificación, como regresión logística, maquinas de vectores de soporte y arboles de decisión.
- Ejemplos de SCIKIT-LEARN
- Fortalezas y debilidades de los clasificadores.

¡Siempre
hacia lo alto!





Clasificadores de SCIKIT-LEARN

En machine learning un algoritmo de clasificación solo es apropiado ***para una tarea*** problemática...difícilmente para dos o más.



Clasificadores de SCIKIT-LEARN

Viejos métodos y poco prácticos.



Perceptrón
ADALINE

Nuevos métodos y muy prácticos.



SCIKIT-LEARN

¡Siempre
hacia lo alto!



Clasificadores de SCIKIT-LEARN

paso a paso para hacer una IA

1. Conocer el dataset: usando **panda.dataframe()**
 - Conocer la Información de la data (tipo de datos por características) **data.info()** ó **data.describe()**
 - Existen datos nulos? **Data.isnull()** ó **data.isnull().sum()**
 - Conocer las dimensiones (filasXcolumnas): **data.shape**
 - Visualizar las primeras filas de la data **data.head()**
2. Dividir los datos (entrenamiento / testeo)
3. Escalado y normalización de valores
4. Entrenar modelo.
5. Realizar predicciones
6. Valorar el modelo.
7. *Visualizar resultados (opcional)*



Que otros modelo existen?

TALLER PARA CASA: ver videos sobre:

1. Regresión lineal (simple/múltiple)
2. Regresión logística y probabilidades condicionales.
3. Vectores de soporte.
4. Árboles de clasificación y regresión
5. K-vecinos (k-means).

Para conocer de forma genérica la funcionalidad de esos modelos



¿Pero qué tipos de modelos de IA Existen?

- ~~Regresión lineal (Es método para encontrar el patrón con una "Mejor Línea de Ajuste")~~
- **Regresión logística.**
- *Árboles de clasificación y regresión.*
- Vectores de soporte
- *K-means*
- *Redes bayesianas*
- Métodos de ensamble de modelos
- **Deep learning**

¡Siempre
hacia lo alto!

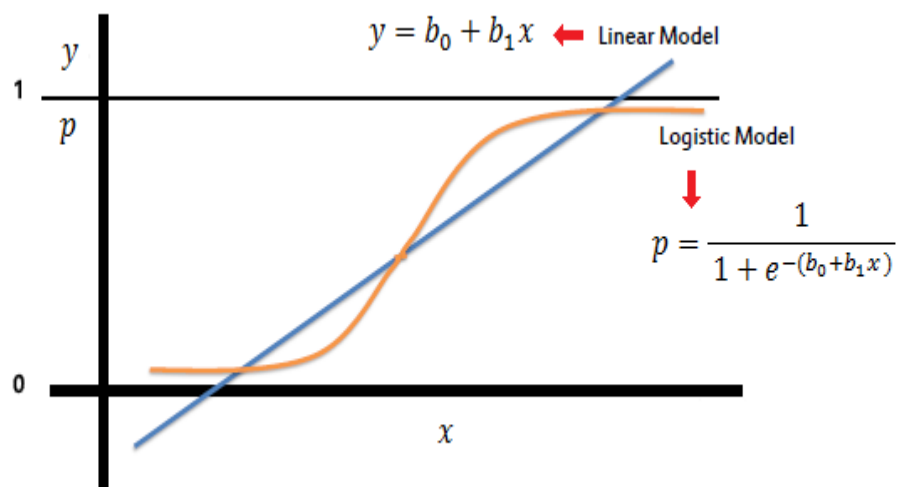


Regresión logística

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

Podemos diferenciar tres tipos de regresiones logísticas:

- **Regresión Logística Binaria:** es la Regresión Logística clásica, en la que hay dos clases a predecir.
- **Regresión Logística Multinomial:** hay más de dos categorías a predecir, pero las clases no guardan ningún orden entre ellas (determinar el texto de un artículo del periódico es de: Entretenimiento, Deportes, Política)
- **Regresión Logística Ordinal:** hay más de dos categorías a predecir y existe un orden entre las categorías (por ejemplo predecir en que posición va a quedar cada equipo al final de la liga de futbol)





Regresión logística

Iniciaremos con la **Regresión Logística Binaria:**

La usaremos para clasificar situaciones con dos posibles estados binarios (1 ó 0)_ “SI/NO” o en un número finito de “etiquetas” o “clases” múltiple.

Algunos Ejemplos de Regresión Logística son:

- Clasificar si el email que llega es Spam o No
- un tumor clasificado como “Benigno” o “Maligno”.



Regresión logística – ejercicio 1: aprobar examen vs horas de estudio

Conociendo el dataset (datos):

El profesor “**Freddy Michael Kruege Myres**”, no tiene claro que tiempo recomendarle a los estudiantes para que estudien para el examen y con ello garantizar las mejores notas de los estudiantes.

Por lo tanto recolecta la base de datos de los últimos 2000 exámenes donde están las horas que estudiaron y si aprobó o reprobó.

La base de datos esta disponible en:

https://github.com/luisFernandoCastellanosG/Machine_learning/tree/master/Databaset_para_trabajar_sklearn

Denominada: **horas_estudio_vs_aprobacion.csv**

Pasos:

1. Importar la librerías
2. Cargar el **dataset** al entorno de trabajo usando **Google drive**
3. Conociendo los datos
4. Prepara los datos de entrenamiento
5. Importa el módulo **LogisticRegression** de la librería **scikit-learn**
6. **Entrenar** la regresión logística con los datos de entrenamiento
7. Usar el modelo entrenado para obtener las **predicciones** con datos nuevos
8. Obtener las **probabilidades** de la predicción

¡Siempre
hacia lo alto!



Regresión logística – ejercicio 1: aprobar examen vs horas de estudio

Pasos:

1. Importar la librerías

```
import numpy as np           #Mejora el soporte para vectores y matrices
import pandas as pd          #Estructura de datos (Ciencia de datos)

import matplotlib.pyplot as plt #Para graficar
import seaborn as sns         #interfaz de alto nivel para dibujar gráficos estadísticos (basada en matplotlib)
```

2. Cargar el **dataset** al entorno de trabajo usando **Google drive**

```
from google.colab import drive
drive.mount('/content/gdrive')
```

2.1. Cargar el dataset en un dataframe de pandas

```
df = pd.DataFrame()
df = pd.read_csv('.../horas_estudio_vs_aprobacion.csv', encoding='utf-8')
```



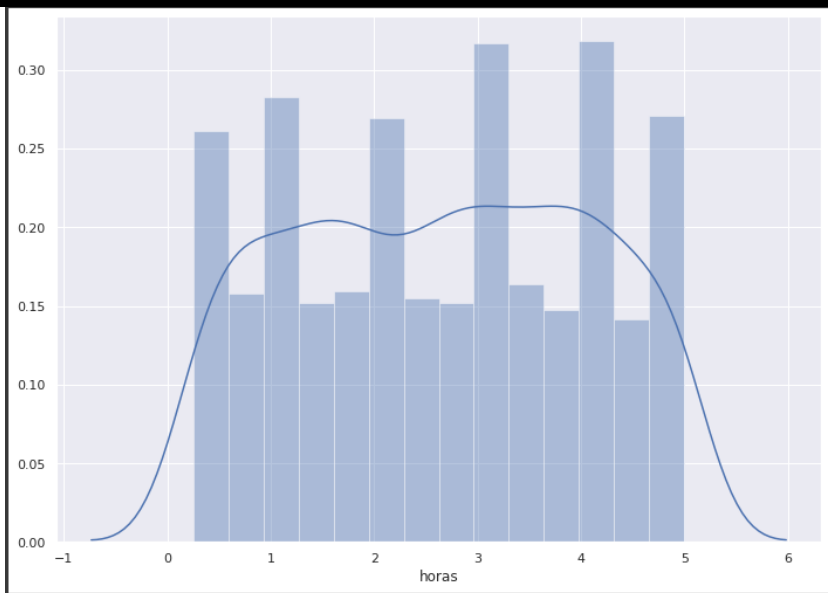
Regresión logística – ejercicio 1: aprobar examen vs horas de estudio

3. Conociendo los datos

```
#primeros 5 registros del dataframe  
df.head()  
#dimensiones del dataframe  
df.shape
```

3.1 visualizando la distribución de los datos.

```
sns.set(rc={'figure.figsize': (11.7, 8.27)}) #tamaño del grafico  
sns.distplot(df['horas']) #agregamos los datos  
plt.show()
```



¡Siempre
hacia lo alto!



Regresión logística – ejercicio 1: aprobar examen vs horas de estudio

4. Separando los datos para el entrenamiento

```
#En X colocaremos el tiempo de estudio que tomaron los estudiantes antes del examen
#apilamos los datos que vienen en 1d a 2d
#opcion 1: usamos np.c_
X = pd.DataFrame(np.c_[df['horas']], columns = ['horas'])
#opcion2: usamos .reshape(-1, 1)
X = np.array(df['horas']).reshape(-1, 1)
#En Y colocaremos el resultado del examen (1 / 0 ) (aprobado / reprobado)
y= np.array(df['aprueba'])
print(X)
```

5. importamos la clase LogisticRegression de scikit-learn.

```
#clase de regresión logística disponible en sklearn
from sklearn.linear_model import LogisticRegression
#Creamos una instancia de la Regresión Logística
regresion_logistica = LogisticRegression()
```

¡Siempre
hacia lo alto!



Regresión logística – ejercicio 1: aprobar examen vs horas de estudio

6. Entrena la regresión logística con los datos de entrenamiento

```
#entrenando  
regresion_logistica.fit(X,y)
```

7. Haciendo predicciones.

Tomaremos una grupo de horas y miraremos que probabilidad de pasar el examen tenemos

```
#definimos que pasa si el estudiante estudia entre 1 a 6 horas  
X_nuevo = np.array([1, 2, 3, 4, 5, 6]).reshape(-1,1)  
#ejecutamos la predicción  
prediccion = regresion_logistica.predict(X_nuevo)  
print(prediccion)
```

¿Que predicción que nos dio es buena?

¡Siempre
hacia lo alto!



Regresión logística – ejercicio 1: aprobar examen vs horas de estudio

8. Generando probabilidades de la predicción

```
probabilidades_prediccion = regresion_logistica.predict_proba(X_nuevo)
#la primera columna es la probabilidad de reprobar
#la segunda columna es la probabilidad de aprobar
print(probabilidades_prediccion)
#si solo nos interesa la probabilidad de aprobar
print(probabilidades_prediccion[:,1])
```

¿Es buena la inferencia de aprobación?

**¡Siempre
hacia lo alto!**



Regresión logística – ejercicio 2: aprobar examen (estudio y tutorías)

Con los mismos pasos pero ahora con una variable nueva:

Horas de estudio + **Horas de tutoría** vs (aprobó / reprobó)

Pasos:

1. Cargando Librerías necesarias

```
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

2. Cargar el dataset en un dataframe de pandas

```
df = pd.DataFrame()
df = pd.read_csv('...../horas_estudio_tutorias_vs_aprobacion.csv', encoding='utf-8')
```





Regresión logística – ejercicio 2: aprobar examen (estudio y tutorías)

3. Conociendo los datos

```
#primeros 5 registros del dataframe  
df.head()
```

3.1 visualizando el tamaño de los datos.

```
#dimensiones del dataframe  
print("matrix df [MxN] ->" + str(df.shape))  
#verificamos que no hayan nulos  
print("---Columnas con valores nulos---")  
print(df.isnull().sum())
```

3.2 visualizando la distribución de los datos.

```
sns.set(rc={'figure.figsize': (11.7, 8.27)}) #tamaño del grafico  
sns.distplot(df['tutorias_mes']) #agregamos los datos  
plt.show()
```



Regresión logística – ejercicio 2: aprobar examen (estudio y tutorías)

4. Separando los datos para el entrenamiento
- Separaremos las características y etiquetamos como **X** e **Y** respectivamente.*
- la variable **X** (horas_autoestudio_diario + tutorías_mes)
 - la variable **Y** (aprobo_perdido)

```
x = df.drop('aprobo_perdido',axis = 1)
y = df.aprobo_perdido
```

Dividiremos los datos en conjuntos de train y test. Esto separará 25%(! valor predeterminado) de los datos en un subconjunto para la parte de prueba y el 75% restante se usará para nuestro subconjunto de entrenamiento.

```
#separamos los datos 25%(test) y 75%(train)
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=4)
#Si deseamos cambiar la proporción solo debemos agregar la variable test_size=0.x
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=4, test_size=0.2)
```

¿De que depende la proporción de datos para entrenamiento?
¿Que hace que se escoja un 25% o un 20% o un 30%?

**¡Siempre
hacia lo alto!**



Regresión logística – ejercicio 2: aprobar examen (estudio y tutorías)

5. importamos la clase LogisticRegression de scikit-learn.

```
#Creamos una instancia de la Regresión Logística  
regresion_logistica = LogisticRegression()
```

6. Entrena la regresión logística con los datos de entrenamiento

```
#entrenando modelo de regresión logística  
regresion_logistica.fit(x,y)
```

7. Haciendo predicciones

```
#usaremos el 25% de los datos para probar el modelo  
#ejecutamos la predicción con datos de prueba (x_test)  
y_prediccion = regresion_logistica.predict(x_test)
```

¡Siempre
hacia lo alto!



Regresión logística – ejercicio 2: aprobar examen (estudio y tutorías)

8. Generando probabilidades de la predicción

```
#comparamos los datos de predicción (y_prediccion) VS los datos de prueba (y_test)
exactitud = metrics.accuracy_score(y_test, y_prediccion)
print("exactitud_porcentaje= "+str(100 * exactitud))
```

probemos con datos manuales para probar el modelo

```
#hagamos prediccion con datos manuales:
#           [horas_estudio , horas_tutoria]
#1-estudiante: [ 4           , 38 ]
#2-estudiante: [ 8           , 29 ]
#3-estudiante: [ 1           , 1  ]
alumnos=np.array([[4, 38], [8, 29],[1,1],[0,0]])
x_nuevo = pd.DataFrame(alumnos,columns = ['horas_autoestudio_diario','tutorias_mes'])
y_prediccion = regresion_logistica.predict(x_nuevo)
print(y_prediccion)
```

¿Tiene lógica la predicción del modelo?

¡Siempre
hacia lo alto!



Tarea de regresión lineal y logísticas:

En la carpeta de github:

https://github.com/luisFernandoCastellanosG/Machine_learning/tree/master/Dataset_para_trabajar_sklearn

Descargar el dataset denominado: **mercadeo_bancario.csv**

El conjunto de datos proviene del [repositorio UCI Machine Learning](#), (*con unos cambios que se le realizaron*) y está relacionado con campañas de marketing directo (llamadas telefónicas) de una institución bancaria portuguesa.

El objetivo de clasificación es **predecir si el cliente se suscribirá (1/0)** a un depósito a plazo (variable y)



Tarea de regresión lineal y logísticas:

Variables del dataset

- **edad** (numérica)
- **trabajo**: tipo de trabajo (categórico: "admin", "housemaid", "management", "retired", "self-employed", "student", "technician", "unemployed", "unknown")
- **conyugal**: estado civil (categórico: "divorciado", "casado", "único", "desconocido")
- **educación** (categórica: "preschool", "primary_school", "technical_school", "Technological_school", "illiterate", "professional.course", "university.degree", "unknown")
- **Total_hijos**: número total de hijos
- **credito_sin_pago**: ¿tiene crédito en incumplimiento de pago? (categóricamente: "no", "sí", "desconocido")
- **prestamo_vivienda**: ¿tiene préstamo de vivienda? (categóricamente: "no", "sí", "desconocido")
- **prestamo_personal**: ¿tiene préstamo personal? (categóricamente: "no", "sí", "desconocido")
- **valor_préstamo**: (numérico) es el valor total del prestamos que tiene con el banco
- **tarjetas_credito**: Número de tarjetas de crédito con otros bancos
- **contacto**: tipo de comunicación de contacto (categórico: "celular", "teléfono")
- **mes**: último mes de contacto del año (categórico: "jan", "feb", "mar", ..., "nov", "dec")
- **dia_semana**: último día de contacto de la semana (categórico: "mon", "tue", "wed", "thu", "fri")
- **duracion_ultimo_contacto**: duración del último contacto, en segundos (numérico). Nota importante: este atributo afecta en gran medida al destino de salida (p. ej., si duration-0 y 'no').
- **campanna**: número de contactos realizados durante esta campaña y para este cliente (numérico, incluye el último contacto)
- **dias_ultimo_contacto**: número de días que pasaron después de que el cliente fue contactado por última vez desde una campaña anterior
- **anterior_contacto**: número de contactos realizados antes de esta campaña y para este cliente (numérico)
- **resultado_anterior**: resultado de la campaña de marketing anterior (categórica: "fracaso", "inexistente", "éxito")
- **numero_empleados**: número de empleados que tiene a su cargo — (numérico)
- **Predecir variable (objetivo deseado)**: (binario: "1", significa "Sí", "0" significa "No")



Tarea de regresión lineal y logísticas:

En grupos de tres personas (A,B,C,D) “*deben darle un nombre al grupo*”, van a crear dos modelos de machine learning:

- 1° modelo, generado con el algoritmo de regresión lineal, donde deben identificar cuales son las variables con mejor correlación y hacer predicción
- 2° modelo, generado con el algoritmo de Regresión Logística Binaria y hacer predicción.

Responder las siguientes preguntas (usando código en Python):

- La edad promedio de los clientes que dijeron que SI y los que dijeron que NO (si_no)
- Cual fue el porcentaje que tomaron para separar el dataset en (train y test) y por que?
- Cual esa la precisión de cada modelo (lineal y logística)?
- En la regresión lineal cuales fueron las variables con mejor correlación y cuales no?

Nota: para realizar predicciones con modelos de machine learning **NUNCA** se debe trabajar con datos en formato **texto**, por lo tanto si los campos están en texto es necesario convertirlos a números, ejemplo el campo “**estado_civil**”

Valores categóricos	Valores en números
married	1
single	2
divorced	3
unknown	0

¡Siempre
hacia lo alto!



Tarea de regresión lineal y logísticas:

Asignación de variables:

Grupo A:

- Edad
- Estado_civil
- Total_hijos
- Contacto
- mes
- Si_no

Grupo B:

- edad
- Trabajo
- Credito_sin_pago
- Valor_préstamo
- dia_semana
- Si_no

Grupo C:

- Edad
- prestamo_vivienda
- dias_ultimo_pago
- anterior_contacto
- duracion_ultimo_contacto
- Si_no

Grupo D:

- Edad
- Educacion
- prestamo_personal
- tarjetas_crédito
- numero_empleados
- Si_no