

Act 11: Programando Regresión Logística en Python

Ana Isabel Loera Gil

26 de marzo del 2025

1 Introducción

Es una técnica de análisis de datos que utiliza las matemáticas para encontrar las relaciones entre dos factores de datos. Luego, utiliza esta relación para predecir el valor de uno de esos factores basándose en el otro. Normalmente, la predicción tiene un número finito de resultados, como un sí o un no.

2 Metodología

2.1 Importar librerías

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from sklearn import linear_model, model_selection
from sklearn.metrics import classification_report,
confusion_matrix, accuracy_score
```

2.2 Leer el archivo csv y mostrar los primeros 5 registros

```
dataframe = pd.read_csv(r"usuarios_win_mac_lin.csv")
print(dataframe.head())
```

2.3 Obtener información estadística básica de nuestro set de datos

```
print(dataframe.describe())
```

2.4 Analizar cuántos usuarios hay de cada tipo de sistema operativo

```
print (dataframe.groupby('clase').size())
```

2.5 Visualización de datos

```
dataframe.drop(['clase'],axis=1).hist()
plt.show()
```

También se puede interrelacionar las entradas de a pares, para ver como se concentran linealmente las salidas de usuarios por colores: Sistema Operativo Windows en azul, Macintosh en verde y Linux en rojo

```
sb.pairplot(dataframe.dropna(), hue='clase',
size=4, vars=["duracion","paginas","acciones","valor"],kind='reg')
plt.show()
```

2.6 Creación del modelo

```
#excluir la columna clase
X=np.array(dataframe.drop(['clase'], axis=1))
#agregar a la columna clase en una nueva variable
y= np.array(dataframe['clase'])
#comprobar la dimension de la matriz
X.shape
```

```
#creacion del modelo de regresion logistica
model = linear_model.LogisticRegression()
model.fit(X,y)
```

```
predictions = model.predict(X)
print(predictions[0:5])
```

```
print(model.score(X, y))
```

2.7 Validación del modelo

Una buena práctica en Machine Learning es la de subdividir nuestro conjunto de datos de entrada en un set de entrenamiento y otro para validar el modelo. Para ello, se dividirán los datos de entrada en forma aleatoria (mezclados) utilizando 80% de registros para entrenamiento y 20% para validar.

```
#validacion del modelo
validation_size = 0.20
seed = 7
X_train, X_validation, Y_train, Y_validation =
model_selection.train_test_split(X,y,test_size=validation_size, random_state=seed)

name='Logistic Regression'
kfold = model_selection.KFold(n_splits=10, shuffle=True ,random_state=seed)
cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold,
```

```

scoring='accuracy')
msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())

print(msg)
predictions = model.predict(X_validation)
print(accuracy_score(Y_validation,predictions))

```

2.8 Reporte de resultados

```

#reporte de resultados
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation,predictions))

```

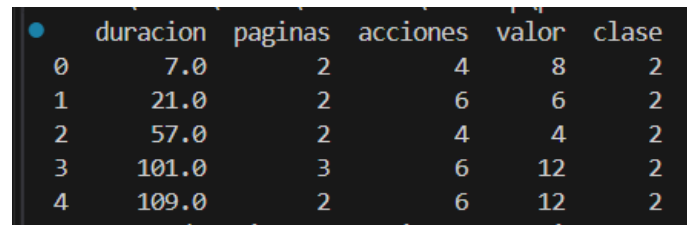
2.9 Clasificación de nuevos valores

```

#clasificacion de nuevos valores
X_new= pd.DataFrame({'duracion':[10], 'paginas': [3], 'acciones':[5],
'valor':[9]})
print(model.predict(X_new))

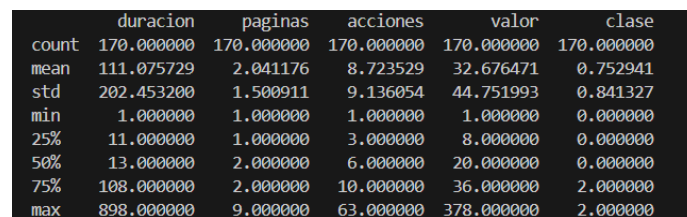
```

3 Resultados



	duracion	paginas	acciones	valor	clase
0	7.0	2	4	8	2
1	21.0	2	6	6	2
2	57.0	2	4	4	2
3	101.0	3	6	12	2
4	109.0	2	6	12	2

Figure 1: Primeros 5 registros de nuestro archivo csv



	duracion	paginas	acciones	valor	clase
count	170.000000	170.000000	170.000000	170.000000	170.000000
mean	111.075729	2.041176	8.723529	32.676471	0.752941
std	202.453200	1.500911	9.136054	44.751993	0.841327
min	1.000000	1.000000	1.000000	1.000000	0.000000
25%	11.000000	1.000000	3.000000	8.000000	0.000000
50%	13.000000	2.000000	6.000000	20.000000	0.000000
75%	108.000000	2.000000	10.000000	36.000000	2.000000
max	898.000000	9.000000	63.000000	378.000000	2.000000

Figure 2: Estadística básica del set de datos

En la siguiente imagen se puede ver la agrupacion de los resultados mediante el método group by, donde 0 representa a Windows, 1 a Mac y 2 a Linux

clase	
0	86
1	40
2	44

Figure 3: Agrupación de los resultados

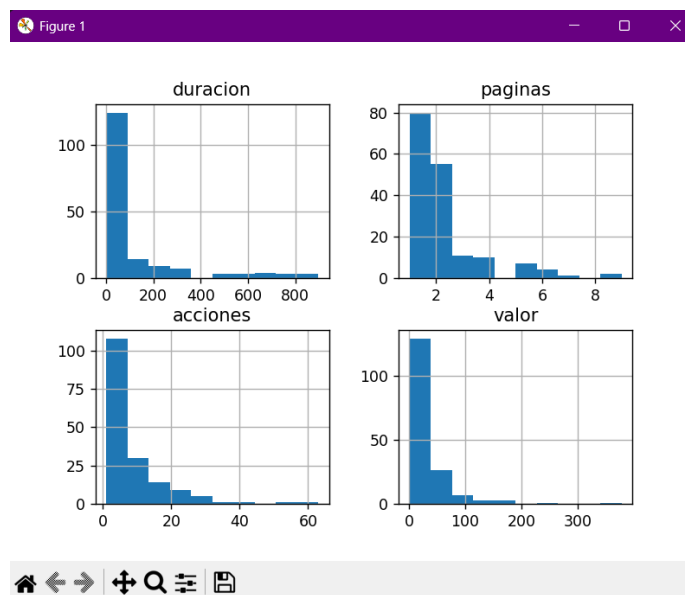


Figure 4: Gráficas de los 4 features de entrada

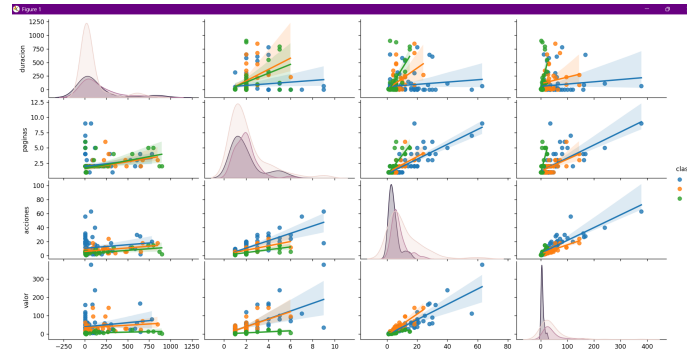


Figure 5: Gráficas con las entradas interrelacionadas a pares

`[2 2 2 2 2]`

Figure 6: Clasificación de todo nuestro conjunto de entradas X utilizando el método “predict(X)”

`0.7823529411764706`

Figure 7: Precisión media de las predicciones

`Logistic Regression: 0.712637 (0.146407)`

Figure 8: Calculo del modelo con el 80% de los datos

`0.8529411764705882`

Figure 9: Cross-validation

`[[16 0 2]
[3 3 0]
[0 0 10]]`

Figure 10: Reporte de los resultados del modelo

	precision	recall	f1-score	support
0	0.84	0.89	0.86	18
1	1.00	0.50	0.67	6
2	0.83	1.00	0.91	10
accuracy			0.85	34
macro avg	0.89	0.80	0.81	34
weighted avg	0.87	0.85	0.84	34

Figure 11: Reporte de los resultados del modelo con el conjunto de validación

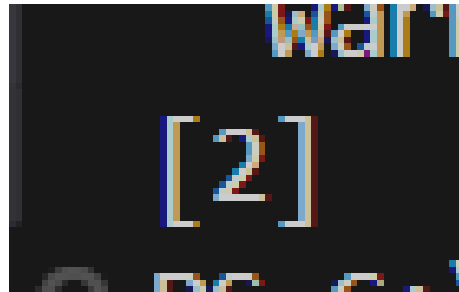


Figure 12: Clasificación de nuevos valores, donde 2 representa a Linux

4 Conclusión

Con este ejemplo pude ver como es que funcionaba una regresión logística en python, aplicado al caso de la clasificación del sistema operativo que utilizaban los usuarios a partir de la navegación de una página de internet. También se pudo hacer uso de la validación del modelo, entrenándolo primero con el 80% y el otro 20% para validar.

5 Referencias bibliográficas

<https://aws.amazon.com/es/what-is/logistic-regression/>
 Ignacio Bagnato, J. (2020). Aprende machine learning. Leanpub.