# Actividad 10

Jose Manuel Enriquez

22 de marzo del 2025

## 1 Introduction

### 1.1 ¿Qué es la regresión logistica?

Es un tipo de modelo estadístico (también conocido como modelo logit) se utiliza a menudo para la clasificación y el análisis predictivo. La regresión logística estima la probabilidad de que ocurra un evento, como votar o no votar, en función de un conjunto de datos determinado de variables independientes.

Dado que el resultado es una probabilidad, la variable dependiente está limitada entre 0 y 1. En la regresión logística, se aplica una transformación logit a las probabilidades, es decir, la probabilidad de éxito dividida por la probabilidad de fracaso. Esto también se conoce comúnmente como probabilidades logarítmicas, o el logaritmo natural de probabilidades

## 2 Metodología

Este ejercicio fue realizado en una Jupyter Notebook y Python 3.12.4 Lo primero que se hizo fue importar todas las librerías necesarias.

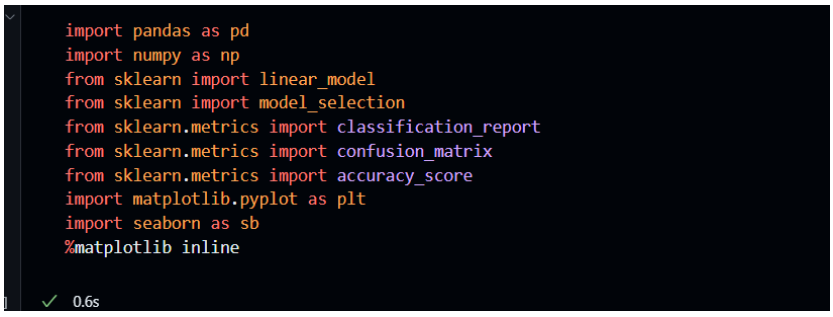


Figure 1: Librerias

Luego importamos el archivos con los datos y vemos su contenido.

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

✓ 0.0s

| | 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 2: Dataframe

```
dataframe.describe()
```
✓ 0.0s

| | 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 |

```
print(dataframe.groupby('clase').size())
```
✓ 0.0s

```
clase
0    86
1    40
2    44
dtype: int64
```

Figure 3: Describe y Groupby

Luego se crean histogramas de los datos

```
dataframe.drop(['clase'], axis = 1).hist()
plt.show()
```
✓ 0.3s

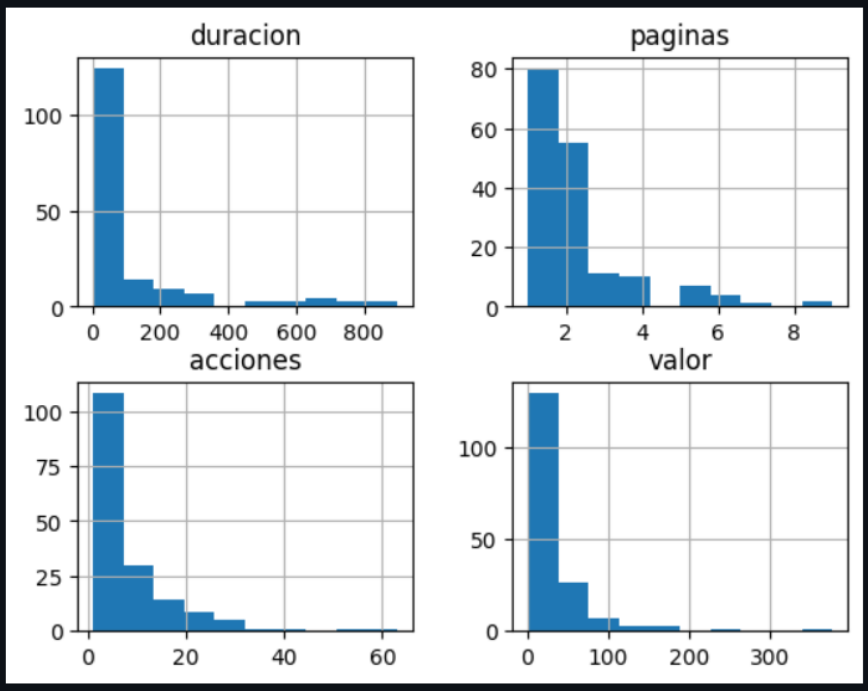


Figure 4: Histograma

Luego creamos los modelos de regresión

```
model = linear_model.LogisticRegression(max_iter=1000)
model.fit(X, y)

predictions = model.predict(X)
print(predictions)

model.score(X, y)
```
✓ 0.2s

Figure 5: Logistic Regression

```
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)
✓ 0.0s
                                                    Generate   + Code   + Markdown

name = 'Logistic Regression'
kfold = model_selection.KFold(n_splits = 10, random_state=seed, shuffle=True)
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)
✓ 3.0s
```

Figure 6: Modelo

# 3 Resultados

La puntuación del modelo fue de 0.77, ahora cuando separamos los datos de entrada con los de validación dio 0.71 y usando "cross validation set" dio como resultado 0.85

La confusion matrix es la siguiente:

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

Figure 7: Confussion Matrix

Y el reporte de clasificación es el siguiente:

| | 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 8: Classification Report

# 4 Conlusiones

Me pareció muy interesante realizar las regresiones con Python ya que no lo había hecho antes de esta manera.