Actividad 11 - Inteligencia Artificial
Programando Regresión Logística en Python

Nombre: Idalia Alejandra Delgado Moreno
Grupo: 31

Introducción
¿Qué es una regresión lógistica

La regresión logística 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.

Metodolodgía

Para realizar esta actividad, hay que seguir las intrucciones que vienen en el libro proporcionado por el profesor de la págia 47 a la 56 del pdf.

1. Importamos las librerías necesarias para el análisis:

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

# Configuración para gráficos
%matplotlib inline
```

2. Cargamos el dataset y visualizamos los primeros registros:

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

3. Generamos estadísticas básicas del dataset:

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

4. Visualizamos la distribución de datos con histogramas:

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

5. Generamos un gráfico de pares para explorar relaciones entre variables:

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

6. Preparamos los datos para el modelo:

```
X = np.array(dataframe.drop(['clase'], axis=1))
y = np.array(dataframe['clase'])
print(X.shape)
```

7. Entrenamos un modelo de Regresión Logística:

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

8. Hacemos predicciones y evaluamos el modelo:

```
predictions = model.predict(X)
print(predictions[:5])
print("Accuracy:", model.score(X, y))
```

9. Dividimos los datos en entrenamiento y validación:

```
validation_size = 0.20
seed = 7
X_train, X_validation, Y_train, Y_validation =
train_test_split(X, y, test_size=validation_size, random_state=seed)
```

10. Validación cruzada del modelo:

```
name = 'Logistic Regression'
model = LogisticRegression()
kfold = KFold(n_splits=10, random_state=7, shuffle=True)
cv_results = 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)
```

11. Entrenamos el modelo con los datos de entrenamiento:

```
model = LogisticRegression()
model.fit(X_train, Y_train)
```

12. Hacemos predicciones y evaluamos con los datos de validación:

```
predictions = model.predict(X_validation)

print(accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
```

13. Hacemos una predicción con nuevos datos:

```
X_new = pd.DataFrame({'duracion': [10], 'paginas': [3],
'acciones': [5], 'valor': [9]})
model.predict(X_new)
```

Resultados:

A continuación, se muestran lo generado por el código:

```
Precisión: 0.8235294117647058
Matriz de Confusión:
[[16  0  2]
 [ 4  2  0]
 [ 0  0 10]]
Reporte de Clasificación:
              precision    recall  f1-score   support

           0       0.80      0.89      0.84        18
           1       1.00      0.33      0.50         6
           2       0.83      1.00      0.91        10

    accuracy                           0.82        34
   macro avg       0.88      0.74      0.75        34
weighted avg       0.85      0.82      0.80        34
```

Figure 1: Resultados.

Conclusión:

En este trabajo, desarrollamos un modelo de Regresión Logística en Python para clasificar el

sistema operativo de los usuarios a partir de su comportamiento de navegación en un sitio web. A través del análisis exploratorio, la visualización de datos y la implementación del modelo, observamos cómo las características de los usuarios pueden utilizarse para predecir su sistema operativo con cierto nivel de precisión. Este enfoque puede extenderse a una amplia variedad de problemas en los que se requiere predecir valores discretos, como la detección de spam, el diagnóstico de enfermedades o la segmentación de clientes. Además, también aprendimos qué tipo de regresión usar dependiendo el tipo de datos que tengamos.