Contenido

[Contexto 2](#_Toc87883154)

[Objetivo 2](#_Toc87883155)

[Fuente de datos 2](#_Toc87883156)

[Preparación del entorno de ejecución 3](#_Toc87883157)

[1) Importar las bibliotecas necesarias 3](#_Toc87883158)

[2) Importar los datos 3](#_Toc87883159)

[Selección de características 3](#_Toc87883160)

[Evaluación visual 3](#_Toc87883161)

[Matriz de correlaciones 5](#_Toc87883162)

[Definición de variables predictoras (X) y variable clase (Y) 5](#_Toc87883163)

[Aplicación del algoritmo 6](#_Toc87883164)

[Regresión logística 6](#_Toc87883165)

[Se hace la división de los datos 7](#_Toc87883166)

[Entrando el modelo 7](#_Toc87883167)

[Validación del modelo 8](#_Toc87883168)

[Modelo de clasificación 9](#_Toc87883169)

[Nuevos pronósticos 9](#_Toc87883170)

[Conclusiones 10](#_Toc87883171)

# Contexto

Objetivo: Clasificar registros clínicos de tumores malignos y benignos de cáncer de mama a partir de imágenes digitalizadas.

Diagrama

Descripción generada automáticamente

## Fuente de datos

Estudios clínicos a partir de imágenes digitalizadas de pacientes con cáncer de mama de Wisconsin (WDBC, Wisconsin Diagnostic Breast Cancer).

Texto

Descripción generada automáticamente

**Fuente:** <https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)>

Registros clínicos de cáncer de mama a partir de imágenes digitalizadas.

Escala de tiempo

Descripción generada automáticamente

# Preparación del entorno de ejecución

## Importar las bibliotecas necesarias

import pandas as pd               # Para la manipulación y análisis de datos

import numpy as np                # Para crear vectores y matrices n dimensionales

import matplotlib.pyplot as plt   # Para la generación de gráficas a partir de los datos

import seaborn as sns             # Para la visualización de datos basado en matplotlib

%matplotlib inline

## Importar los datos

Fuente de datos: WDBCOriginal.csv

# Si se usa Google Colab

#from google.colab import files

#files.upload()

# Si se importan los datos desde Drive

#from google.colab import drive

#drive.mount('/content/drive')

BCancer = pd.read\_csv("WDBCOriginal.csv")

BCancer

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

print(BCancer.groupby('Diagnosis').size())

# De una población total de 569 muestras, 357 presentan un tipo de cáncer benigno y 212 maligno.

Un conjunto de letras blancas en un fondo azul

Descripción generada automáticamente con confianza media

# Selección de características

## Evaluación visual

Se utiliza una matriz de correlaciones con el propósito de seleccionar variables significativas.

sns.pairplot(BCancer, hue='Diagnosis')

plt.show()

Texto, Tabla, Pizarra

Descripción generada automáticamente

#plt.plot(BCancer['Radius'], BCancer['Perimeter'], 'b+')

sns.scatterplot(x='Radius', y='Perimeter',data=BCancer, hue='Diagnosis')

plt.title('Gráfico de dispersión')

plt.xlabel('Radio')

plt.ylabel('Perímetro')

plt.show()

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

#plt.plot(BCancer['Concavity'], BCancer['ConcavePoints'], 'b+')

sns.scatterplot(x='Concavity', y='ConcavePoints',data=BCancer, hue='Diagnosis')

plt.title('Gráfico de dispersión')

plt.xlabel('Concavity')

plt.ylabel('ConcavePoints')

plt.show()

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

## Matriz de correlaciones

CorrBCancer = BCancer.corr(method='pearson')

CorrBCancer

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

plt.figure(figsize=(14,7))

MatrizInf = np.triu(BCancer.corr())

sns.heatmap(BCancer.corr(), cmap='RdBu\_r', annot=True, mask=MatrizInf)

plt.show()

Gráfico

Descripción generada automáticamenteVariables seleccionadas:

1. Textura [Posición 3]
2. Area [Posición 5]
3. Smoothness [Posición 6]
4. Compactness [Posición 7]
5. Symmetry [Posición 10]
6. FractalDimension [Posición 11]

# Definición de variables predictoras (X) y variable clase (Y)

BCancer = BCancer.replace({'M': 0, 'B': 1})

BCancer

Imagen que contiene azul, firmar, grande, reloj

Descripción generada automáticamente

print(BCancer.groupby('Diagnosis').size())

# 0 = MALIGNO: 212 muestras

# 1 = BENIGNO: 357 muestras

# Variables predictoras

X = np.array(BCancer[['Texture','Area','Smoothness','Compactness','Symmetry', 'FractalDimension']])

# X = BCancer.iloc[:, [3,5,6,7,10,11]].values # iloc para seleccionar filas y columnas según su posición

pd.DataFrame(X)

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

# Variable Clase

Y = np.array(BCancer['Diagnosis'])

pd.DataFrame(Y)

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

plt.figure(figsize=(10,7))

plt.scatter(X[:,0], X[:,1], c = BCancer.Diagnosis)

plt.grid()

plt.xlabel('Texture')

plt.ylabel('Area')

plt.show()

# En morado Cáncer maligno

# En amarillo Cáncer benigno

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

# Aplicación del algoritmo

## Regresión logística

# Se importan las bibliotecas necesarias

from sklearn import linear\_model # Para la regresión lineal / pip install scikit-learn

from sklearn import model\_selection

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

X\_train, X\_validation, Y\_train, Y\_validation = model\_selection.train\_test\_split(X, Y, test\_size=0.2, random\_state=1234, shuffle=True)

# Datos de entrenamiento: 70, 75 u 80% de los datos

# Datos de prueba: 20, 25 o 30% de los datos

### Se hace la división de los datos

pd.DataFrame(X\_train)

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

pd.DataFrame(Y\_train)

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

### Entrando el modelo

# Se entrena el modelo a partir de los datos de entrada

Clasificacion = linear\_model.LogisticRegression() # Se crea el modelo

Clasificacion.fit(X\_train, Y\_train) # Se entrena el modelo

# Predicciones probabilísticas

Probabilidad = Clasificacion.predict\_proba(X\_train)

pd.DataFrame(Probabilidad)

Pantalla azul con letras blancas

Descripción generada automáticamente con confianza media

# Predicciones probabilísticas de los datos de prueba

Probabilidad = Clasificacion.predict\_proba(X\_validation)

pd.DataFrame(Probabilidad) # A partir de las probabilidades se hacen el etiqueta de si es cancerígeno o no

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

# Predicciones con clasificación final

Predicciones = Clasificacion.predict(X\_validation)

pd.DataFrame(Predicciones) # A partir de las probabilidades obtenidas anteriormente

# se hace el etiquetado de si es cancerígeno o no

Interfaz de usuario gráfica

Descripción generada automáticamente

# Se calcula la exactitud promedio de la validación

Clasificacion.score(X\_validation, Y\_validation)

# Se tiene un 93% de exactitud este modelo: 0.9385964912280702

# Validación del modelo

# Matriz de clasificación

Y\_Clasificacion = Clasificacion.predict(X\_validation)

Matriz\_Clasificacion = pd.crosstab(Y\_validation.ravel(), Y\_Clasificacion, rownames=['Real'], colnames=['Clasificación'])

Matriz\_Clasificacion

# El modelo se equivocó en un total de 7 casos, de los cuales 6 fueron falsos positivos y 1 falso negativo

# Para los casos benignos, se equivocó 6 de 68 veces.

# Para los casos malignos, se equivocó 1 de 39 veces.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteTabla

Descripción generada automáticamente

# Reporte de clasificación

print(classification\_report(Y\_validation, Y\_Clasificacion))

print("Exactitud", Clasificacion.score(X\_validation, Y\_validation))

print("Precisión: ",classification\_report(Y\_validation, Y\_Clasificacion).split()[10])

print("Tasa de error: ",1-Clasificacion.score(X\_validation, Y\_validation))

print("Sensibilidad: ",classification\_report(Y\_validation, Y\_Clasificacion).split()[11])

print("Especificidad: ",classification\_report(Y\_validation, Y\_Clasificacion).split()[6])

# Precisión positiva del 92% (92% de exactitud al clasificar casos de cáncer benigno)

# Precisión negativa del 97% (97% de exactitud al clasificar casos de cáncer maligno)

# En general, el modelo es muy bueno, ya que tiene casi 94% de exactitud. (93.86%)

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

# Modelo de clasificación

# Ecuación del modelo

print("Intercept:",Clasificacion.intercept\_)

print("Coeficientes:\n",Clasificacion.coef\_)

print("--------------------------------------------------------------------------------")

print("Prob = 1/1+𝑒^−(𝑎+𝑏𝑋))")

print("Ecuación del modelo: ")

print("a + bX =",Clasificacion.intercept\_[0],"+",Clasificacion.coef\_[0][0],"(Texture) +",Clasificacion.coef\_[0][1],"(Area) +"

,Clasificacion.coef\_[0][2],"(Smoothness) +",Clasificacion.coef\_[0][3],"(Compactness) +",Clasificacion.coef\_[0][4],"(Symmetry) +"

,Clasificacion.coef\_[0][5],"(FractalDimension)")

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

## Nuevos pronósticos

# Paciente P-842302 (1) Tumor maligno

PacienteID1 = pd.DataFrame({'Texture':[10.38],

                            'Area':[1001.0],

                            'Smoothness':[0.11840],

                            'Compactness':[0.27760],

                            'Symmetry':[0.2419],

                            'FractalDimension':[0.07871]})

Clasificacion.predict(PacienteID1)

De acuerdo con nuestro modelo y con los datos ingresados, este paciente tiene un tumor maligno.

# Paciente P-92751 (569) Tumor benigno

PacienteID2 = pd.DataFrame({'Texture':[24.54],

                            'Area':[181.0],

                            'Smoothness':[0.05263],

                            'Compactness':[0.04362],

                            'Symmetry':[0.1587],

                            'FractalDimension':[0.05884]})

Clasificacion.predict(PacienteID2)

De acuerdo con nuestro modelo y con los datos ingresados, este paciente tiene un tumor benigno.

# Conclusiones

En esta práctica, a través de registros clínicos de cáncer de mama tomados de imágenes digitalizadas de la WDBC (Wisconsin Diagnostic Breast Cancer), se pudo hacer un análisis de estos datos, esto gracias a la aplicación del algoritmo de regresión logística, que pertenece a la categoría de aprendizaje supervisado, el cual su principal objetivo es predecir valores binarios (en este caso, el objetivo fue predicir si el paciente tiene un tipo de tumor maligno o benigno)

Los resultados obtenidos del algoritmo fueron los siguientes:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Predicción** | |
|  |  | **Positivos** | **Negativos** |
| **Observación** | **Positivos** | Verdaderos Positivos (VP): **68** | Falsos Negativos (FN):  **1** |
| **Negativos** | Falsos Positivos (FP): **6** | Verdaderos Negativos (VN):  **39** |

Esto nos indica que el modelo se equivocó en un total de 7 casos, de los cuales 6 fueron falsos positivos y 1 falso negativo.

* Para los casos benignos, se equivocó 6 de 68 veces.
* Para los casos malignos, se equivocó 1 de 39 veces.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. **Exactitud (Acurracy)**

El 93.86% de los datos fueron clasificados correctamente.

1. **Tasa de error (Misclassification Rate)**

El 6.14% de los datos fueron clasificados incorrectamente.

1. **Precisión (Precision)**

El 91.89% de los datos positivos fueron clasificados correctamente.

1. **Sensibilidad (Recall, Sensitivity, True Positive Rate)**

El 98.55% de los datos positivos totales fueron clasificados correctamente.

1. **Especificidad (Especificity, True Negative Rate)**

El 86.66% de los datos negativos totales fueron clasificados correctamente.

**Modelo logístico:**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Ecuación del modelo: