INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
Reporte Práctica 7 "Algoritmos de clasificación"
Integrantes: Cruz Barragan Ryan Nathanael
Rodríguez Vázquez Joshua Levi
Torres Abonce Luis Miguel
Correos: RNCB0963@gmail.com
joshualevirv@gmail.com
luiseishon9@gmail.com

I. Resumen

Los algoritmos de clasificación en el aprendizaje automático son esenciales para diversas aplicaciones, que van desde el reconocimiento de imágenes hasta la detección de correos electrónicos no deseados.

El objetivo principal de esta práctica es utilizar el lenguaje de MATLAB para poder realizar dichos algoritmos de clasificación, los cuales veremos más adelante en la práctica.

El desarrollo de la práctica también incluye dichos códigos así como la explicación de dichos códigos para poder realizar la clasificación mencionada anteriormente.

II. Palabras clave

- **MATLAB**: plataforma de programación y cálculo numérico utilizada para analizar datos, desarrollar algoritmos y realizar cálculos numéricos.
- **Algoritmos de clasificación:** son métodos en el campo del aprendizaje automático que se utilizan para asignar categorías o etiquetas a datos de entrada.
- **Regresión logística:** es un tipo de algoritmo de clasificación utilizado en el aprendizaje automático que modela la probabilidad de que una instancia pertenezca a una clase particular.

III. Introducción

En el ámbito del aprendizaje de máquinas, los algoritmos de clasificación juegan un papel fundamental en una variedad de aplicaciones, desde el reconocimiento de imágenes hasta la detección de spam en correos electrónicos. En esta práctica, nos adentraremos en el mundo de estos algoritmos para comprender su funcionamiento y aplicabilidad en diferentes contextos.

Exploramos algunos de los algoritmos de clasificación más destacados, como la Regresión Logística, las Máquinas de Vectores de Soporte (SVM), los Árboles de Decisión, los Bosques Aleatorios (Random Forest), Naive Bayes, Redes Neuronales y los Métodos de Ensamblaje. Cada uno de estos algoritmos tiene sus propias fortalezas y debilidades, y seleccionar el adecuado depende del problema específico, la naturaleza de los datos y los recursos disponibles.

En este contexto, crearemos un entorno de aprendizaje donde podremos entrenar y evaluar modelos utilizando datos sintéticos y conjuntos de datos reales. Tomaremos como ejemplo la implementación de un modelo de Regresión Logística y Naive Bayes en MATLAB, desde la preparación de los datos hasta la evaluación del rendimiento del modelo.

A través de esta práctica, desarrollaremos una comprensión sólida de cómo estos algoritmos funcionan en la práctica, así como su aplicación en diferentes escenarios del mundo real en el campo del aprendizaje automático.

IV. Desarrollo

Primero empezaremos haciendo un algoritmo de regresión lineal utilizando MATLAB, lo primero que deberemos hacer es preparar los datos, lo cual se puede realizar de la siguiente manera:

Pseudocodigo:
% Generar datos sintéticos
inicializar generador de números aleatorios con semilla 0 % Para reproducibilidad
n = 100 % Número de muestras
X = concatenar(matriz de números aleatorios normales (n, 2) + 1, matriz de números aleatorios normales (n, 2) - 1) % Características
y = concatenar(vector de unos (n, 1), vector de ceros (n, 1)) % Etiquetas

% Visualizar los datos
crear nueva figura

```
dibujar dispersión de gráficos usando X(:,1), X(:,2) y etiquetas y, con colores 'r' y 'b', y marcadores 'x' y 'o'
etiquetar eje X como 'Característica 1'
etiquetar eje Y como 'Característica 2'
agregar título 'Datos sintéticos para regresión logística'
```

En matlab, entrenar este tipo de modelos es relativamente sencillo, lo que debemos usar es la función fitglm de la siguiente manera:

```
% Convertir los datos a una tabla
data = array2table(X, 'VariableNames', {'Feature1', 'Feature2'});
data.Response = y;
% Ajustar el modelo de regresión logística
model = fitglm(data, 'Response ~ Feature1 + Feature2', 'Distribution', 'binomial');
% Mostrar el resumen del modelo
disp(model);
```

Después, podemos realizar las predicciones con estos modelos en nuevos datos o en los mismos datos que utilizamos para el entrenamiento, de la siguiente manera:

```
Pseudo codigo:
% Convertir los datos a una tabla
datos = convertir_a_tabla(X, 'NombresVariables', {'Caracteristica1', 'Caracteristica2'})
datos.Respuesta = y

% Ajustar el modelo de regresión logística
modelo = ajustar_regresion_logistica(datos, 'Respuesta ~ Caracteristica1 + Caracteristica2', 'Distribucion', 'binomial')

% Mostrar el resumen del modelo
mostrar(modelo)

Después, podemos realizar las predicciones con estos modelos en nuevos datos o en los mismos datos que utilizamos para el entrenamiento, de la siguiente manera:

% Predecir las probabilidades
probabilidades = predecir(modelo, datos)

% Convertir probabilidades a etiquetas (0 o 1) con un umbral de 0.5
predicciones = redondear(probabilidades)

% Evaluar la precisión del modelo
precision = promedio(predicciones == y)
```

```
imprimir('Precisión del modelo: %.2f%%\n', precision * 100)

% Visualizar las predicciones
crear nueva figura
dibujar dispersión de gráficos usando X(:,1), X(:,2) y predicciones, con colores 'r' y 'b', y marcadores 'x' y 'o'
etiquetar eje X como 'Característica 1'
etiquetar eje Y como 'Característica 2'
agregar título 'Predicciones del modelo de regresión logística'
```

Ahora, para el segundo ejemplo, implementaremos el algoritmo de Naive Bayes, utilizando datos sintéticos y utilizando el lenguaje de programación Matlab.

Lo primero que debemos de realizar, como el ejemplo pasado es la preparación de los datos:

```
% Generar datos sintéticos
inicializar generador de números aleatorios con semilla 1 % Para reproducibilidad
n = 100 % Número de muestras por clase
X = concatenar(matriz de números aleatorios normales (n, 2) + 1, matriz de números aleatorios normales (n, 2) - 1) % Características
y = concatenar(vector de unos (n, 1), vector de ceros (n, 1)) % Etiquetas: 1 y 0

% Visualizar los datos
crear nueva figura
dibujar dispersión de gráficos usando X(:,1), X(:,2) y etiquetas y, con colores 'r' y 'b', y marcadores 'x' y 'o'
etiquetar eje X como 'Característica 1'
etiquetar eje Y como 'Característica 2'
agregar título 'Datos sintéticos para Naive Bayes'
```

Entrenar utilizando naive bayes es relativamente sencillo en Matlab, basta con llamar una sola función llamada “fitcnb” la cual podemos ver a continuación:

```
Pseudocodigo:
% Entrenar el modelo Naive Bayes
modeloNaiveBayes = entrenar_naive_bayes(X, y)

% Mostrar el modelo entrenado
mostrar(modeloNaiveBayes)
```

Y listo, podemos realizar predicciones utilizando el modelo entrenado anteriormente de la siguiente manera:

```
% Predecir las etiquetas
predicciones = predecir(modeloNaiveBayes, X)

% Evaluar la precisión del modelo
precision = promedio(predicciones == y)
imprimir('Precisión del modelo: %.2f%%\n', precision * 100)

% Visualizar las predicciones
d = 0.02 % Densidad de la malla
[x1Malla, x2Malla] = generar_malla(min(X(:,1)):d:max(X(:,1)),
min(X(:,2)):d:max(X(:,2)))
xMalla = concatenar_columnas(x1Malla(:), x2Malla(:))
prediccionesMalla = predecir(modeloNaiveBayes, xMalla)

crear nueva figura
dibujar dispersión de gráficos usando X(:,1), X(:,2) y etiquetas y, con colores 'r' y 'b', y marcadores 'x' y 'o'
mantener gráfica actual

dibujar dispersión de gráficos usando xMalla(:,1), xMalla(:,2) y prediccionesMalla, con colores 'r' y 'b', y marcadores '.'

etiquetar eje X como 'Característica 1'
etiquetar eje Y como 'Característica 2'
agregar título 'Naive Bayes con frontera de decisión'
agregar leyenda 'Clase 1', 'Clase 0', 'Ubicaciones predichas'
liberar gráfica actual
```

V. Resultados

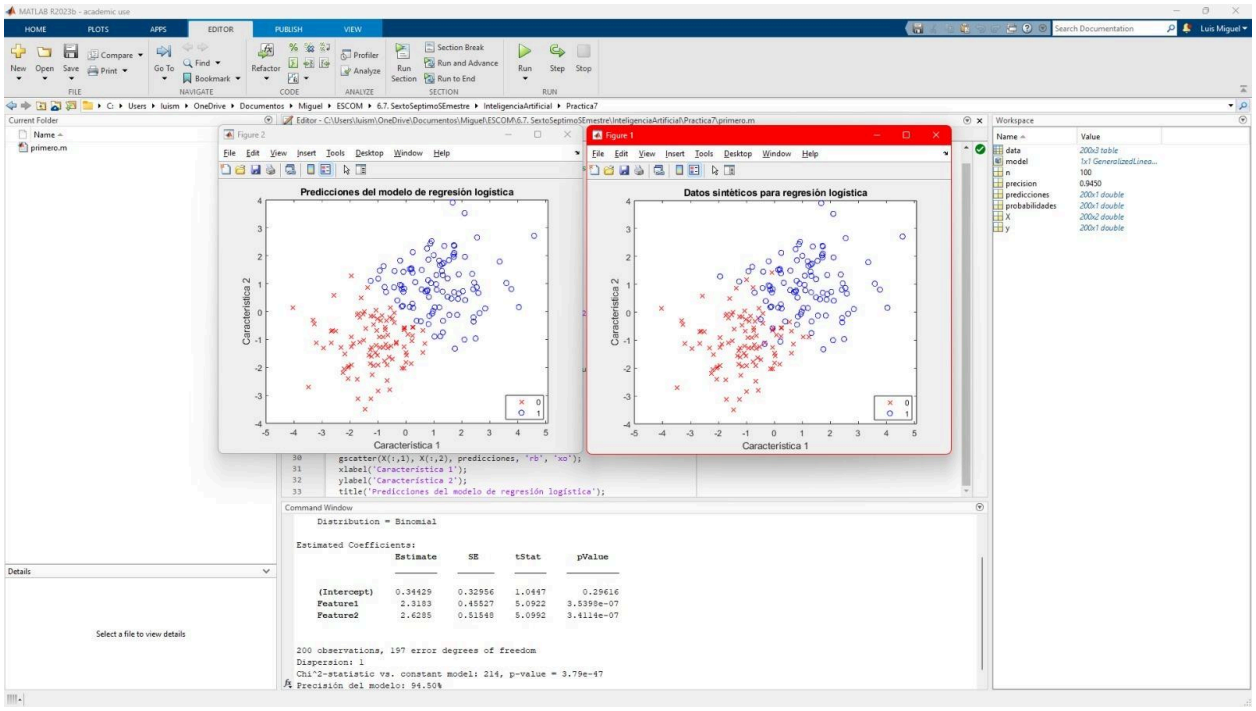

Figura 1. Primer programa de Matlab.

En el primer programa podemos ver que al correrlo se generan dos ventanas, la primera muestra las predicciones del modelo y el segundo los datos estadísticos, estos están basados en la información y el código que fue mostrado anteriormente.

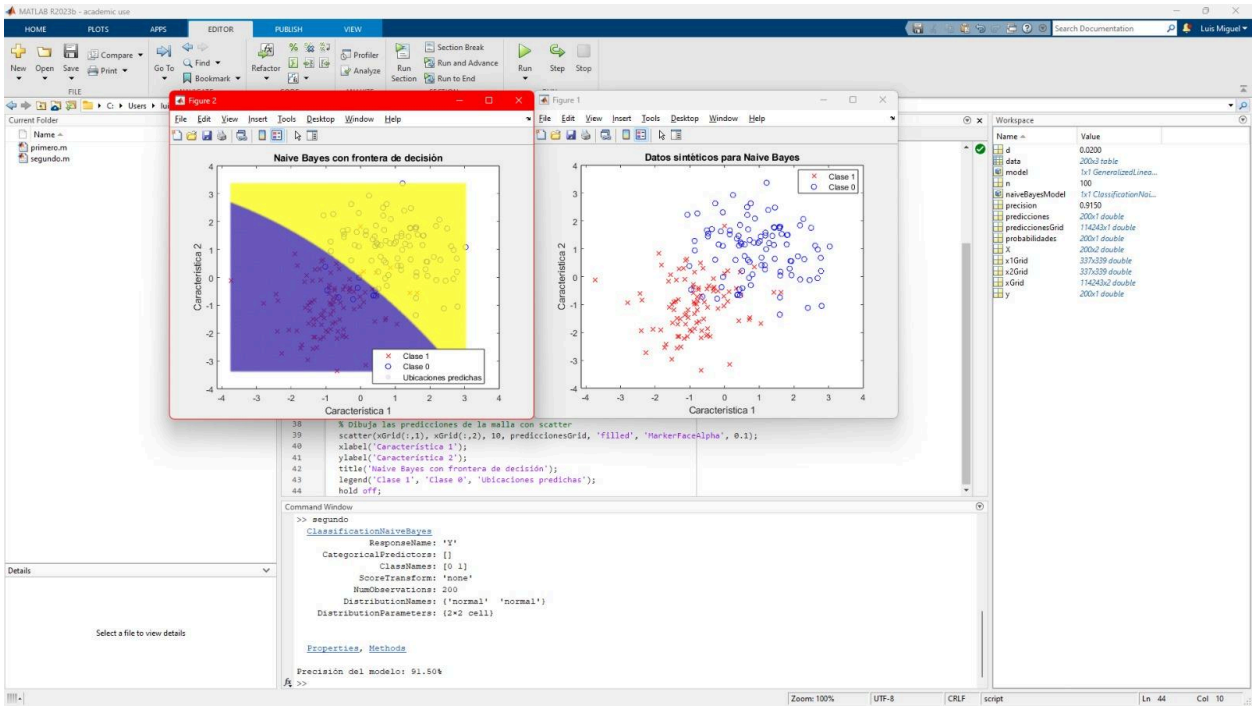

Figura 2. Segundo programa de Matlab.

En el segundo programa podemos observar de la misma manera que al correrlo se generan dos ventanas, una que muestra la frontera de decisión la cual separa los datos, y en la segunda ventana vemos los datos sintéticos para el algoritmo de clasificación de Naive Bayes.

VI. Conclusión

En esta práctica, hemos explorado los algoritmos de clasificación en el contexto del aprendizaje automático, específicamente implementando los modelos de Regresión Logística y Naive Bayes en MATLAB. A lo largo de esta práctica, hemos demostrado cómo preparar y visualizar datos, entrenar modelos de clasificación y evaluar su rendimiento mediante métricas de precisión y visualización de predicciones.

El uso de MATLAB ha facilitado la implementación de estos algoritmos, permitiéndonos enfocarnos en entender los conceptos fundamentales y los pasos prácticos para llevar a cabo la clasificación. Los ejemplos presentados muestran cómo estos modelos pueden ser aplicados a datos sintéticos, proporcionando una base sólida para abordar problemas de clasificación en datos reales.

Hemos aprendido que la Regresión Logística es un modelo sencillo y eficiente para problemas de clasificación binaria, mientras que Naive Bayes, aunque basado en una suposición de independencia a menudo irrealista, es rápido y eficaz para grandes conjuntos de datos. Ambos algoritmos tienen sus propias ventajas y limitaciones, y la

elección entre ellos dependerá de las características específicas del problema que estemos tratando de resolver.

VII. Bibliografia

[1] Colaboradores de Wikipedia. (2024b, abril 4). Clasificador bayesiano ingenuo. Wikipedia, la Enciclopedia Libre. https://es.wikipedia.org/wiki/Clasificador_bayesiano_ingenuo