INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

INTELIGENCIA ARTIFICIAL

PRÁCTICA 9

CLASIFICADORES KNN Y NAIVE BAYES

NOMBRE DEL ALUMNO: GARCÍA QUIROZ GUSTAVO IVAN

NOMBRE DEL PROFESOR: GARCÍA FLORIANO ANDRES

GRUPO: 6CV3

FECHA DE ENTREGA DEL REPORTE: 22/10/2024

Índice

1 Introducción

La clasificación es una tarea que consiste en asignar una categoría o etiqueta a un conjunto de datos de entrada. Existen diversos algoritmos de clasificación que se utilizan para abordar este tipo de problemas de clasificación, cada uno con sus propias características y enfoques. En esta práctica, nos enfocaremos en la implementación y evaluación de dos clasificadores clásicos: el Clasificador Euclidiano y el Clasificador 1-NN (1-Nearest Neighbor).

El Clasificador Euclidiano se basa en el cálculo de la distancia euclidiana entre un punto de prueba y los puntos de entrenamiento, asignando la clase del vecino más cercano. Por otro lado, el Clasificador 1-NN extiende este concepto al considerar los k vecinos más cercanos, aunque en esta práctica utilizaremos específicamente k=1.

Para el desarrollo de esta práctica, se utilizó el lenguaje de programación Python. Python es una opción popular en el campo del aprendizaje automático y el análisis de datos debido a su sintaxis clara, la amplia disponibilidad de bibliotecas y herramientas de soporte, y su facilidad de uso.

Para evaluar el desempeño de estos clasificadores, se implementarán tres métodos de validación comúnmente utilizados: Holdout 70/30, Validación Cruzada de 10 Pliegues y Leave-One-Out. Estos métodos permiten estimar la capacidad de generalización de los modelos y brindar una visión más completa de su rendimiento.

En esta práctica, se analizan los resultados del desempeño de los clasificadores en tres conjuntos de datos diferentes. Estos tres conjuntos de datos presentan diferentes niveles de complejidad y características, lo que permitirá evaluar el desempeño de los clasificadores en diversos escenarios. Estos tres conjuntos de datos son: flores iris, diagnóstico de cáncer de mama y vinos de diferentes regiones.

Iris Plant Dataset es un conjunto de datos clásico en el campo del aprendizaje automático, que contiene información sobre tres especies de flores iris. Cada muestra está descrita por cuatro características (longitud y anchura de los sépalos y pétalos) y pertenece a una de las tres clases (tipos) de iris. En total, este conjunto de datos cuenta con 150 muestras.

Breast Cancer Dataset un conjunto de datos proviene de exámenes de diagnóstico de cáncer de mama. Cada muestra representa las características de un tumor, y la tarea es clasificar si el tumor es benigno o maligno. El conjunto de datos cuenta con alrededor de 569 muestras.

Wine Dataset es un conjunto de datos contiene información química sobre vinos de diferentes regiones. El objetivo es clasificar los vinos en uno de los tres tipos (clases) diferentes. El conjunto de datos consta de 178 muestras.

2 Marco teórico

2.1 Conjunto de datos de la flor iris

El conjunto de datos flor Iris o conjunto de datos iris de Fisher es un conjunto de datos multivariante introducido por Ronald Fisher en su artículo de 1936, The use of multiple measurements in taxonomic problems (El uso de medidas múltiples en problemas taxonómicos) como un ejemplo de análisis discriminante lineal. A veces, se llama Iris conjunto de datos de Anderson porque Edgar Anderson coleccionó los datos para cuantificar la variación morfológica de la flor Iris de tres especies relacionadas.

Figura 1 Planta Iris.

2.1.1 Características del conjunto de datos de la flor iris

El conjunto de datos contiene 50 muestras de cada una de tres especies de Iris (Iris setosa, Iris virginica e Iris versicolor). Se midió cuatro rasgos de cada muestra: el largo y ancho del sépalo y pétalo, en centímetros. Basado en la combinación de estos cuatro rasgos, Fisher desarrolló un modelo discriminante lineal para distinguir entre una especie y otra.

	sepal_length	sepal_width	petal_length	petal_width	species
114	5.8	2.8	5.1	2.4	virginica
62	6.0	2.2	4.0	1.0	versicolor
33	5.5	4.2	1.4	0.2	setosa
107	7.3	2.9	6.3	1.8	virginica
7	5.0	3.4	1.5	0.2	setosa

Figura 2 Características del conjunto de datos de la flor iris.

2.2 Conjunto de datos de diagnóstico de cáncer de mama en Wisconsin

El conjunto de datos de diagnóstico de cáncer de mama de Wisconsin es una reconocida colección de datos que se utiliza ampliamente en el aprendizaje automático y la investigación médica. Este conjunto de datos, que se origina a partir de imágenes digitalizadas de aspiraciones con aguja fina (AAF) de masas mamarias, facilita el análisis de las características de los núcleos celulares para ayudar en el diagnóstico del cáncer de mama.

El conjunto de datos de diagnóstico de cáncer de mama de Wisconsin es un conjunto de datos conocido que se utiliza habitualmente en el aprendizaje automático. El conjunto de datos fue seleccionado por el Dr. William H. Wolberg, W. Nick Street y Olvi L. Mangasarian. Contiene características calculadas a partir de imágenes digitalizadas de muestras de tejido mamario aspirado con aguja fina (FNA).

Breast Cancer Wisconsin (Diagnostic) Dataset

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890	0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902	0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758	0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300	0
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678	0

Figura 3 Conjunto de datos de diagnóstico de cáncer de mama en Wisconsin.

2.2.1 Características del conjunto de datos del cáncer de mama de Wisconsin

- Número de instancias: 569.
- Número de atributos: 30 atributos numéricos utilizados para la predicción, junto con una etiqueta de clase.
- Distribución de clases: 212 - Maligno, 357 – Benigno.

El conjunto de datos incluye 30 características, entre ellas la media, el error estándar y los valores "peores" o "más grandes", calculados para cada imagen. Estas características encapsulan varios aspectos de las características de los núcleos celulares:

- Radio medio: Media de las distancias desde el centro a los puntos del perímetro.
- Textura media: desviación estándar de los valores de la escala de grises.
- Perímetro medio: Perímetro del tumor.
- Área media: Área del tumor.
- Suavidad media: variación en las longitudes de los radios.
- Compacidad media: Perímetro^2 / Área - 1.0.
- Concavidad media: Severidad de las porciones cóncavas del contorno.
- Puntos cóncavos medios: Número de porciones cóncavas del contorno.
- Simetría media: Simetría de los núcleos celulares.
- Dimensión fractal media: "Aproximación de la línea de costa" - 1

Clases	2
Muestras por clase	212(M),357(B)
Muestras totales	569
Dimensionalidad	30
Características	real, positivo

Tabla 1 Características del conjunto de datos del cáncer de mama de Wisconsin.

2.3 Conjunto de datos del vino

El conjunto de datos Wine Recognition es un conjunto de datos de referencia clásico ampliamente utilizado en el aprendizaje automático para tareas de clasificación. Proporciona información valiosa sobre la clasificación del vino en función de varios atributos químicos.

El conjunto de datos original de Wine fue creado por Forina, M. et al, como parte del proyecto PARVUS, un paquete extensible para exploración, clasificación y correlación de datos, realizado en el Instituto de Análisis y Tecnologías Farmacéuticas y Alimentarias, Génova, Italia.

El conjunto de datos sobre vinos contiene los resultados de un análisis químico de vinos cultivados en tres regiones diferentes de Italia. En concreto, incluye 13 atributos derivados de mediciones de diversos componentes presentes en los vinos. Estos atributos suelen incluir factores como el contenido de alcohol, los niveles de acidez y las concentraciones de diferentes compuestos químicos, como fenoles y flavonoides. Estos atributos proporcionan información valiosa sobre la composición química de los vinos y pueden utilizarse para tareas de clasificación de vinos.

2.3.1 Características del conjunto de datos del vino

El conjunto de datos de reconocimiento de Wine posee varias características clave que lo hacen ideal para tareas de clasificación y experimentación con aprendizaje automático. Estas características brindan información sobre la estructura, el tamaño y la naturaleza de los datos que contiene el conjunto de datos.

Número de instancias:	178
Número de atributos:	13 atributos numéricos, predictivos y la clase
Información del atributo:	• Alcohol • Ácido málico • Ceniza • Alcalinidad de la ceniza

	• Magnesio • Fenoles totales • Flavonoides • Fenoles no flavonoides • Proantocianinas • Intensidad del color • Matiz • OD280/OD315 de vinos diluidos • Prolina

Tabla 2 Características del conjunto de datos del vino.

Tres clases correspondientes al origen del vino:

1. Clase 1: Vinos de la primera región (denominados como "clase_0")
2. Clase 2: Vinos de la segunda región (denominados como "clase_1")
3. Clase 3: Vinos de la tercera región (denominados como "clase_2")

El conjunto de datos de reconocimiento de vinos se utiliza habitualmente para tareas de aprendizaje supervisado, en particular algoritmos de clasificación. Los investigadores y profesionales suelen emplear técnicas de aprendizaje automático para crear modelos que puedan predecir con precisión el origen de los vinos en función de su composición química.

2.4 Algoritmo de K-vecino más cercano (KNN)

El algoritmo K-Nearest Neighbors (KNN) es un método de aprendizaje automático supervisado que se emplea para resolver problemas de clasificación y regresión. Evelyn Fix y Joseph Hodges desarrollaron este algoritmo en 1951, que posteriormente fue ampliado por Thomas Cover. El artículo explora los fundamentos, el funcionamiento y la implementación del algoritmo KNN.

KNN es uno de los algoritmos de clasificación más básicos y esenciales en el aprendizaje automático. Pertenece al dominio del aprendizaje supervisado y encuentra una aplicación intensa en el reconocimiento de patrones, la minería de datos y la detección de intrusiones.

Es ampliamente descartable en escenarios de la vida real, ya que no es paramétrico, lo que significa que no hace suposiciones subyacentes sobre la distribución de datos (a diferencia de otros algoritmos como GMM, que suponen una distribución gaussiana de los datos dados). Se nos proporcionan algunos datos previos (también llamados datos de entrenamiento), que clasifican las coordenadas en grupos identificados por un atributo.

A modo de ejemplo, considere la siguiente figura de puntos de datos que contienen dos características:

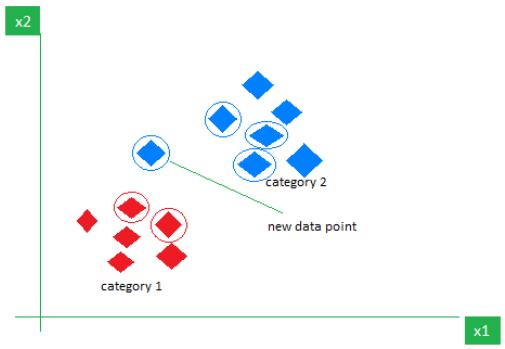

Figura 4 Visualización del funcionamiento del algoritmo KNN.

Ahora, dado otro conjunto de puntos de datos (también llamados datos de prueba), se asigna estos puntos a un grupo analizando el conjunto de entrenamiento.

Métricas de distancia utilizadas en el algoritmo KNN

Como sabemos, el algoritmo KNN nos ayuda a identificar los puntos o grupos más cercanos para un punto de consulta. Pero para determinar los grupos o puntos más cercanos para un punto de consulta necesitamos algunas métricas. Para este propósito, utilizamos las siguientes métricas de distancia:

2.4.1 Distancia euclidiana

Esta no es otra cosa que la distancia cartesiana entre los dos puntos que se encuentran en el plano/hiperplano. La distancia euclidiana también se puede visualizar como la longitud de la línea recta que une los dos puntos en consideración. Esta métrica nos ayuda a calcular el desplazamiento neto realizado entre los dos estados de un objeto.

$$\text{distancia}\left(x, incógnita_i\right) = \sqrt{\sum_{j=1}^{d}\left(x_{yo} - X_{i_{yo}}\right)^2]}$$

Figura 5 Distancia euclidiana.

2.4.2 Distancia de Manhattan

La métrica de distancia de Manhattan se utiliza generalmente cuando nos interesa la distancia total recorrida por el objeto en lugar del desplazamiento. Esta métrica se calcula sumando la diferencia absoluta entre las coordenadas de los puntos en n dimensiones.

$$d\left(x, y\right) = \sum_{yo=1}^{norte} \left| x_i - y_i \right|$$

Figura 6 Distancia de Manhattan.

2.4.3 Distancia de Minkowski

Podemos decir que la distancia euclidiana, así como la distancia de Manhattan, son casos especiales de la distancia de Minkowski.

$$d(x, y) = (\sum_{yo=1}^{norte} (x_i - y_i)^{pag})^{\frac{1}{pag}}$$

Figura 7 Distancia de Minkowski.

De la fórmula anterior podemos decir que cuando p = 2 entonces es la misma que la fórmula para la distancia euclidiana y cuando p = 1 entonces obtenemos la fórmula para la distancia de Manhattan.

Las métricas analizadas anteriormente son las más comunes cuando se trata de un problema de aprendizaje automático, pero también existen otras métricas de distancia, como la distancia de Hamming, que resultan útiles cuando se tratan problemas que requieren comparaciones superpuestas entre dos vectores cuyo contenido puede ser tanto booleano como de cadena.

2.5 Algoritmo de K-vecino más cercano (KNN)

Los métodos Bayes ingenuos son un conjunto de algoritmos de aprendizaje supervisado basado en la aplicación del teorema de Bayes con la suposición "ingenua" de independencia condicional entre cada par de características dada la valor de la variable de clase. El teorema de Bayes establece lo siguiente relación, variable de clase dada y característica dependiente vector a través de , :yx1xn

$$P(y \mid x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n \mid y)}{P(x_1, \dots, x_n)}$$

Utilizando la ingenua suposición de independencia condicional de que

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y),$$

para todos, esta relación se simplifica ai

$$P(y \mid x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^{n} \quad P(x_i \mid y)}{P(x_1, \dots, x_n)}$$

Dado que es constante dada la entrada, Podemos utilizar la siguiente regla de clasificación: $P(x_1, \dots, x_n)$

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

y podemos usar la estimación Máxima A Posteriori (MAP) para estimar y ; El primero es entonces la frecuencia relativa de la clase en el conjunto de entrenamiento. $P(x_i \mid y)$

Los diferentes clasificadores Bayes ingenuos difieren principalmente por los supuestos que hacer con respecto a la distribución de $P(x_i \mid y)$.

A pesar de sus suposiciones aparentemente demasiado simplificadas, el ingenuo Bayes Los clasificadores han funcionado bastante bien en muchas situaciones del mundo real, como es famoso por Clasificación de documentos y filtrado de spam. Requieren una pequeña cantidad de los datos de entrenamiento para estimar los parámetros necesarios. (Para fines teóricos) razones por las que el Bayes ingenuo funciona bien, y en qué tipos de datos lo hace.

Los aprendices y clasificadores de Bayes ingenuos pueden ser extremadamente rápidos en comparación con más métodos sofisticados. El desacoplamiento de las distribuciones de características condicionales de la clase significa que cada La distribución puede estimarse de forma independiente como una distribución unidimensional. Esto, a su vez, ayuda a aliviar los problemas derivados de la maldición de dimensionalidad.

Por otro lado, aunque el ingenuo Bayes es conocido como un clasificador decente, Se sabe que es un mal estimador, por lo que los resultados de probabilidad no deben tomarse demasiado en serio.

2.5.1 Gaussiano ingenuo de Bayes

La función GaussianNB implementa el algoritmo gaussiano ingenuo de Bayes para clasificación. Se supone que la probabilidad de las características es gaussiana:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Los parámetros se estiman utilizando la máxima verosimilitud de $\sigma_y \mu_y$.

3 Desarrollo de la práctica.

El desarrollo de esta práctica se centró en Implementar y evaluar diferentes clasificadores (Naive Bayes y K-Nearest Neighbors) utilizando tres conjuntos de datos clásicos: Iris, Breast Cancer y Wine, mediante distintos métodos de validación.

Para esta práctica de clasificación el proceso se diseñó con el propósito de evaluar los clasificadores Naive Bayes y K-Nearest Neighbors (KNN) utilizando conjuntos de datos estándar en ciencia de datos.

El código se estructuró en torno a tres procesos: la selección de datasets, la implementación de clasificadores y la aplicación de las técnicas de validación. Esta aproximación permite obtener el análisis sobre el comportamiento y las capacidades predictivas de los modelos bajo diferentes condiciones.

3.1 Conjuntos de Datos

La elección de los datasets constituyó un elemento crítico en la metodología. Se seleccionaron tres conjuntos de datos: Iris, Breast Cancer y Wine. Cada uno de estos datasets presenta características únicas que permiten evaluar el rendimiento de los clasificadores desde múltiples perspectivas.

El dataset de Iris, con sus características morfológicas de flores, ofrece un problema de clasificación simple. El dataset de Breast Cancer proporciona un contexto médico complejo, con implicaciones directas en la detección temprana de patologías. El conjunto de datos de Wine permite analizar la clasificación de categorías de productos de vino.

3.2 Implementación de los Clasificadores

Los clasificadores se implementaron siguiendo un protocolo de selección y configuración meticuloso. Se utilizaron dos familias de algoritmos con enfoques fundamentalmente diferentes:

- Naive Bayes (Distribución Gaussiana): Un clasificador probabilístico basado en el teorema de Bayes, que asume independencia entre las características.
- K-Nearest Neighbors (KNN): Un método de aprendizaje supervisado que clasifica basándose en la proximidad de las instancias en el espacio de características.

Para el KNN, se exploraron tres configuraciones de vecinos (K=3, K=5, K=7) para comprender cómo la variación en este hiperparámetro influye en la precisión del modelo.

3.3 Técnicas de Validación

La práctica incorporó tres técnicas de validación:

1. Hold Out (70/30): División tradicional de datos con énfasis en la capacidad de generalización.
2. 10-Fold Cross-Validation: Método que divide los datos en 10 subconjuntos para una evaluación estadísticamente más sólida.
3. Leave-One-Out: Técnica que utiliza cada instancia como conjunto de prueba, proporcionando una validación detallada.

3.4 Métricas de Evaluación

La evaluación del rendimiento se realizó mediante dos métricas:

- Accuracy: Proporción de predicciones correctas sobre el total de predicciones.
- Matriz de Confusión: Herramienta de visualización que desglosa los resultados por clase, permitiendo un análisis profundo de los errores de clasificación.

3.5 Implementación del Código

3.5.1 Carga de Datasets

La función load_datasets() constituye el punto de entrada para la obtención de los conjuntos de datos. Utiliza las funciones predefinidas de Scikit-learn para cargar tres datasets:

```python
def load_datasets():
    #Carga los tres datasets solicitados
    datasets = {
        'Iris': load_iris(),
        'Breast Cancer': load_breast_cancer(),
        'Wine': load_wine()
    }
    return datasets
```

Figura 8 Función para carga datasets.

3.5.2 Creación de Clasificadores

La función create_classifiers() instancia los diferentes algoritmos de clasificación con configuraciones predeterminadas:

```python
def create_classifiers():
    #Crea los clasificadores con sus configuraciones
    classifiers = {
        'Naive Bayes': GaussianNB(),
        'KNN-3': KNeighborsClassifier(n_neighbors=3),
        'KNN-5': KNeighborsClassifier(n_neighbors=5),
        'KNN-7': KNeighborsClassifier(n_neighbors=7)
    }
    return classifiers
```

Figura 9 Función para crear algoritmos de clasificación.

3.6 Métodos de Validación

En esta sección se detalla la implementación de tres métodos de validación distintos: Hold Out 70/30, 10-Fold Cross-Validation y Leave-One-Out. Estos métodos se implementaron dentro de la función evaluate_classifier, que acepta como parámetros el clasificador a evaluar, los datos de entrada, y el método de validación a utilizar.

3.6.1 Hold Out 70/30

El método Hold Out divide el conjunto de datos en dos partes: 70% para entrenamiento y 30% para prueba. Este método realiza una única división de los datos, entrena el modelo con el conjunto de entrenamiento y evalúa su desempeño con el conjunto de prueba. Se utiliza estado aleatorio igual a 42 para asegurar la reproducibilidad de los resultados.

```python
def holdout_validation(classifier, X, y):
    #Implementa la validación Hold Out 70/30
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
    classifier.fit(X_train, y_train)
    y_pred = classifier.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)
    conf_matrix = confusion_matrix(y_test, y_pred)

    return accuracy, conf_matrix
```

Figura 10 Función para Hold Out.

3.6.2 10-Fold Cross-Validation

La validación cruzada de 10 pliegues divide el conjunto de datos en 10 partes iguales, utilizando 9 partes para entrenamiento y 1 para prueba en cada iteración. Este método realiza 10 iteraciones de entrenamiento y prueba, donde cada muestra del conjunto de datos es utilizada exactamente una vez como dato de prueba. Las matrices de confusión de cada iteración se suman para obtener una matriz de confusión general del proceso.

```python
def kfold_validation(classifier, X, y, n_folds=10):
    #Implementa la validación 10-Fold Cross-Validation
    kf = KFold(n_splits=n_folds, shuffle=True, random_state=42)
    accuracies = []
    conf_matrices = []

    for train_index, test_index in kf.split(X):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]

        classifier.fit(X_train, y_train)
        y_pred = classifier.predict(X_test)

        accuracies.append(accuracy_score(y_test, y_pred))
        conf_matrices.append(confusion_matrix(y_test, y_pred))

    mean_accuracy = np.mean(accuracies)
    std_accuracy = np.std(accuracies)
    total_conf_matrix = sum(conf_matrices)

    return mean_accuracy, std_accuracy, total_conf_matrix
```

Figura 11 Método para la validación cruzada de 10 pliegues.

3.6.3 Leave-One-Out

El método Leave-One-Out es un caso especial de validación cruzada donde el número de pliegues es igual al número de muestras en el conjunto de datos. En este método, cada muestra se utiliza una vez como dato de prueba mientras que el resto de las muestras se utilizan para entrenamiento. Se mantienen listas separadas para almacenar todas las predicciones y valores verdaderos, lo que permite calcular la matriz de confusión general al final del proceso.

El código se realizó utilizando LeaveOneOut de scikit-learn:

```python
def leave_one_out_validation(classifier, X, y):
    #Implementa la validación Leave-One-Out
    loo = LeaveOneOut()
    accuracies = []
    all_predictions = []
    all_true = []

    for train_index, test_index in loo.split(X):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]

        classifier.fit(X_train, y_train)
        y_pred = classifier.predict(X_test)

        all_predictions.extend(y_pred)
        all_true.extend(y_test)
        accuracies.append(1 if y_pred == y_test else 0)

    mean_accuracy = np.mean(accuracies)
    conf_matrix = confusion_matrix(all_true, all_predictions)

    return mean_accuracy, conf_matrix
```

Figura 12 Método para Leave-One-Out

El código completo de estos métodos de validación se integra en la función principal. Esta función coordina la evaluación de cada combinación de clasificador y método de validación sobre los tres conjuntos de datos seleccionados, proporcionando una estructura organizada para la comparación sistemática de los resultados.

- Iris Plant Dataset.
- Breast Cancer Dataset.
- Wine Dataset.

4 Análisis de Resultados

Este análisis se dividirá por conjuntos de datos, observando el rendimiento de ambos clasificadores con los diferentes métodos de validación.

Las siguientes tablas muestran un resumen de los valores obtenidos en cada clasificador y en cada metodo de validación. Es importante destacar que los métodos de validación no son comparables entre sí, por ejemplo, el resultado de aplicar Hold Out no puede compararse con el resultado de aplicar Leave One Out, o K Fold Cross Validation.

	Naive bayes			**KNN (K=5)**		
	HO	KF CV	LOO	HO	KF CV	LOO
Iris	0.9415	0.96	0.9533	1	0.9733	0.9667
Breast Cancer	0.9357	0.9384	0.9385	0.9591	0.9332	0.9367
Wine	1	0.9778	0.9775	0.7407	0.6641	0.6966

Figura 13 Tabla de valores de Accuracy

Naive bayes

HO

Iris

	Setosa	Versicolor	Virginica
Setosa	19	0	0
Versicolor	0	12	1
Virginica	0	0	13

Breast Cancer

	P	N
P	57	6
N	4	104

Wine

	Clase 1	Clase 2	Clase 3
Clase 1	19	0	0
Clase 2	0	21	0
Clase 3	0	0	14

KF CV

Iris

	Setosa	Versicolor	Virginica
Setosa	50	0	0
Versicolor	0	47	3
Virginica	0	3	47

Breast Cancer

	P	N
P	189	23
N	12	345

Wine

	Clase 1	Clase 2	Clase 3
Clase 1	57	2	0
Clase 2	0	69	2
Clase 3	0	0	48

LOO

Iris

	Setosa	Versicolor	Virginica
Setosa	50	0	0
Versicolor	0	47	3
Virginica	0	4	46

Breast Cancer

	P	N
P	189	23
N	12	345

Wine

	Clase 1	Clase 2	Clase 3
Clase 1	57	2	0
Clase 2	0	69	2
Clase 3	0	0	48

Tabla 3 Tabla de valores de las Matrizes de confusión (Euclidiano).

K-NN (K=5)

HO

Iris

	Setosa	Versicolor	Virginica
Setosa	19	0	0
Versicolor	0	13	0
Virginica	0	0	13

Breast Cancer

	P	N
P	57	6
N	1	107

Wine

	Clase 1	Clase 2	Clase 3
Clase 1	17	0	2
Clase 2	1	15	5
Clase 3	1	5	8

KF CV

Iris

	Setosa	Versicolor	Virginica
Setosa	50	0	0
Versicolor	0	47	3
Virginica	0	1	49

Breast Cancer

	P	N
P	188	24
N	14	343

Wine

	Clase 1	Clase 2	Clase 3
Clase 1	57	1	5
Clase 2	5	47	19
Clase 3	6	24	18

LOO

Iris

	Setosa	Versicolor	Virginica
Setosa	50	0	0
Versicolor	0	47	3
Virginica	0	2	48

Breast Cancer

	P	N
P	190	22
N	14	343

Wine

	Clase 1	Clase 2	Clase 3
Clase 1	52	1	6
Clase 2	6	49	16
Clase 3	6	19	23

Tabla 4 Tabla de valores de las Matrizes de confusión (K-NN).

4.1 Resultados

Conjunto de datos de la flor iris

```
==================================================
Dataset: Iris
==================================================

Clasificador: Naive Bayes
------------------------------

Hold Out 70/30:
Accuracy: 0.9778
Matriz de Confusión:
[[19  0  0]
 [ 0 12  1]
 [ 0  0 13]]
10-Fold Cross-Validation:
Accuracy promedio: 0.9600 (±0.0533)
Matriz de Confusión:
[[50  0  0]
 [ 0 47  3]
 [ 0  3 47]]
Leave-One-Out:
Accuracy: 0.9533
Matriz de Confusión:
[[50  0  0]
 [ 0 47  3]
 [ 0  4 46]]
Clasificador: KNN-5
------------------------------

Hold Out 70/30:
Accuracy: 1.0000
Matriz de Confusión:
[[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
```

```
10-Fold Cross-Validation:
Accuracy promedio: 0.9733 (±0.0442)
Matriz de Confusión:
[[50  0  0]
 [ 0 47  3]
 [ 0  1 49]]
Leave-One-Out:
Accuracy: 0.9667
Matriz de Confusión:
[[50  0  0]
 [ 0 47  3]
 [ 0  2 48]]
```

Figura 14 Resultados del conjunto de datos de flores iris

Conjunto de datos de diagnóstico de cáncer de mama.

```
==================================================
Dataset: Breast Cancer
==================================================

Clasificador: Naive Bayes
------------------------------

Hold Out 70/30:
Accuracy: 0.9415
Matriz de Confusión:
[[ 57   6]
 [  4 104]]
10-Fold Cross-Validation:
Accuracy promedio: 0.9384 (±0.0319)
Matriz de Confusión:
[[189  23]
 [ 12 345]]
Leave-One-Out:
Accuracy: 0.9385
Matriz de Confusión:
[[189  23]
 [ 12 345]]
Clasificador: KNN-5
------------------------------

Hold Out 70/30:
Accuracy: 0.9591
Matriz de Confusión:
[[ 57   6]
 [  1 107]]
```

```
Leave-One-Out:
Accuracy: 0.9332
Matriz de Confusión:
[[188  24]
 [ 14 343]]
10-Fold Cross-Validation:
Accuracy promedio: 0.9367 (±0.0355)
Matriz de Confusión:
[[190  22]
 [ 14 343]]
```

Figura 15 Resultados de diagnóstico de cáncer de mama.

Conjunto de datos de clasificación de vinos.

```
==================================================
Dataset: Wine
==================================================

Clasificador: Naive Bayes
------------------------------

Hold Out 70/30:
Accuracy: 1.0000
Matriz de Confusión:
[[19  0  0]
 [ 0 21  0]
 [ 0  0 14]]
10-Fold Cross-Validation:
Accuracy promedio: 0.9778 (±0.0369)
Matriz de Confusión:
[[57  2  0]
 [ 0 69  2]
 [ 0  0 48]]
Leave-One-Out:
Accuracy: 0.9775
Matriz de Confusión:
[[57  2  0]
 [ 0 69  2]
 [ 0  0 48]]
Clasificador: KNN-5
------------------------------

Hold Out 70/30:
Accuracy: 0.7407
Matriz de Confusión:
[[17  0  2]
 [ 1 15  5]
 [ 1  5  8]]
```

```
10-Fold Cross-Validation:
Accuracy promedio: 0.6641 (±0.1001)
Matriz de Confusión:
[[53  1  5]
 [ 5 47 19]
 [ 6 24 18]]
Leave-One-Out:
Accuracy: 0.6966
Matriz de Confusión:
[[52  1  6]
 [ 6 49 16]
 [ 6 19 23]]
```

Figura 16 Resultados del conjunto de datos de clasificación de vinos.

5 Conclusiones

La práctica de cada algoritmo de clasificación sirvió para aprender y categorizar información de manera inteligente. Naive Bayes y K-Nearest Neighbors demostraron ser estrategias para abordar problemas de clasificación. Mientras Naive Bayes ofreció una perspectiva probabilística, KNN ofreció un enfoque más intuitivo basado en la proximidad de los datos.

Los diferentes métodos de validación utilizados Hold Out, 10-Fold Cross-Validation y Leave-One-Out fueron fundamentales para garantizar la veracidad de los resultados. Cada técnica aportó una perspectiva única sobre el rendimiento de los modelos, permitiendo una evaluación de métricas de precisión.

Los conjuntos de datos seleccionados Iris, Breast Cancer y Wine representaron diversos resultados de clasificación. Cada dataset reveló fortalezas y limitaciones específicas de los clasificadores, subrayando la importancia de elegir el método adecuado según el contexto.

El uso de métricas como accuracy y la desviación estándar proporcionó una visión estadística del rendimiento. No solo importaba la precisión promedio, sino también la consistencia de los resultados a través de diferentes datos.

La implementación práctica demostró que no existe un clasificador universal. El rendimiento varía significativamente dependiendo del conjunto de datos, la configuración de los parámetros y el método de validación. Esta variación marca la necesidad de la experimentación en el desarrollo de modelos de machine learning.

6 Referencias.

- *Breast cancer Wisconsin (diagnostic) dataset. (2024, April 30). GeeksforGeeks. https://www.geeksforgeeks.org/breast-cancer-wisconsin-diagnostic-dataset/*
- *Improve, G. (2017, April 14). K-nearest neighbor(KNN) algorithm. GeeksforGeeks. https://www.geeksforgeeks.org/k-nearest-neighbours/*
- *sagar99 Follow Improve. (2024, February 28). Iris dataset. GeeksforGeeks. https://www.geeksforgeeks.org/iris-dataset/*
- *Scikit-learn. (n.d.). Scikit-learn.org. Retrieved 12 November 2024, from https://scikit-learn.org/*
- *Wine dataset in sklearn. (2024, May 1). GeeksforGeeks. https://www.geeksforgeeks.org/wine-dataset/*
- H. Zhang (2004). La optimalidad de Naive Bayes. Proc. FLAIRS.

7 Código

-