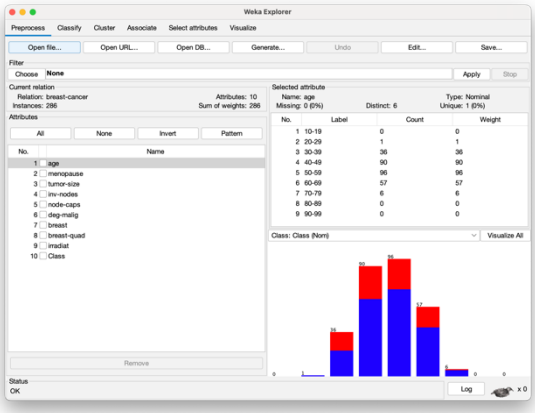
APRENDIZAJE SUPERVIDADO

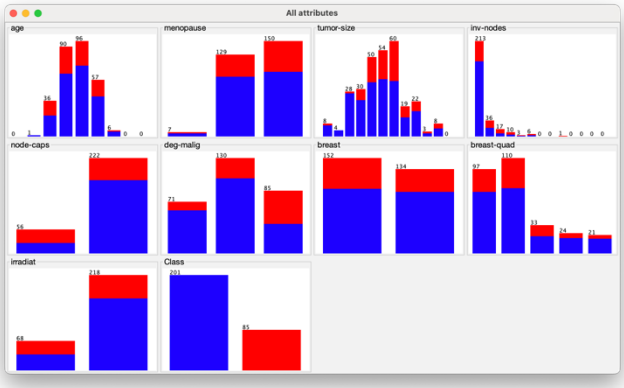
MINERIA DE DATOS 2024

MARTA DE CASTRO LEIRA

1.SELECCIÓN DEL DATASET

- **Dataset**: Breast Cancer (provisto por defecto en Weka)
- **Número de instancias**: 286
- **Número de atributos:** 10
- **Descripción:** Este conjunto de datos contiene información clínica y biológica de tumores de mama, con el objetivo de clasificar los tumores como malignos o benignos
- **Objetivo del análisis:** Predecir la clase (recurrence-events vs no-recurrence-events) aplicando técnicas de preprocesamiento y algoritmos de clasificación





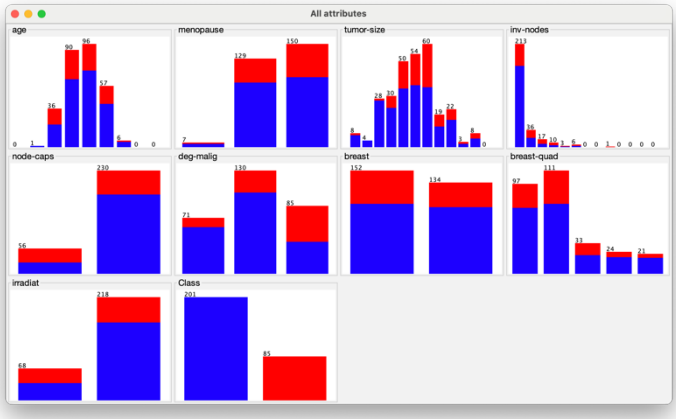
2. TÉCNICAS DE PRPROCESAMIENTO

Dado que la calidad de los datos impacta al rendimiento del modelo, se implementaron las siguientes técnicas de preprocesamiento:

2.1 Manejo de Valores Faltantes

Para abordar los valores faltantes en el conjunto de datos, se utilizó el filtro ***ReplaceMissingValues***, que imputa los datos según el tipo de atributo. En los atributos numéricos, como edad y tamaño del tumor, se reemplazaron los valores faltantes con la media aritmética, lo que permite mantener la distribución general sin introducir sesgos significativos. En los atributos categóricos, como menopausia y node-caps, se utilizó la moda (la categoría más frecuente), asegurando que los datos imputados sean representativos de las características predominantes.

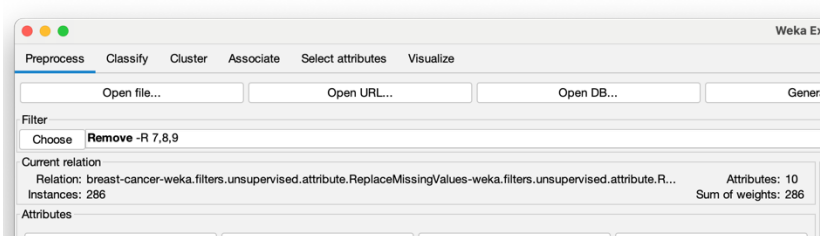
Este proceso es crucial para evitar que los valores faltantes afecten negativamente el entrenamiento del modelo o reduzcan el número de instancias disponibles. La imputación asegura datos completos y consistentes, preservando tanto la calidad como la cantidad del conjunto de datos, lo que mejora la eficacia de los algoritmos de clasificación.

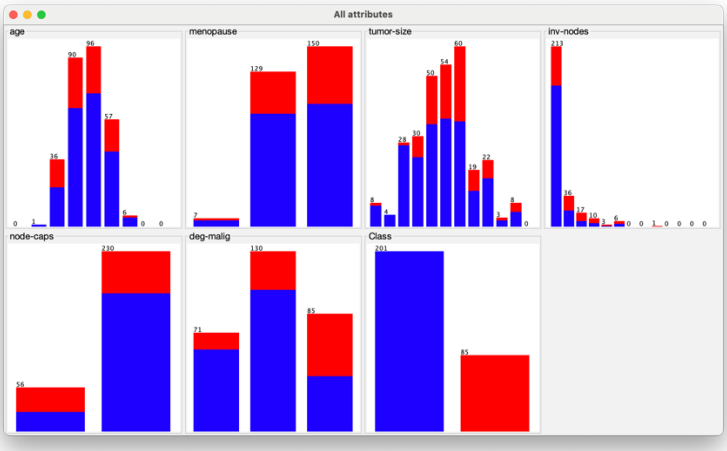


2.2 Eliminación de Atributos

Se aplicó el filtro ***Remove*** para eliminar atributos considerados irrelevantes o redundantes en el contexto del análisis. Entre ellos, el atributo Breast fue descartado, ya que no tiene impacto en la recurrencia del tumor. Asimismo, Breast-quad se excluyó debido a su información redundante, y Irradiat fue eliminado para enfocar el análisis exclusivamente en las características clínicas y biológicas más relevantes.

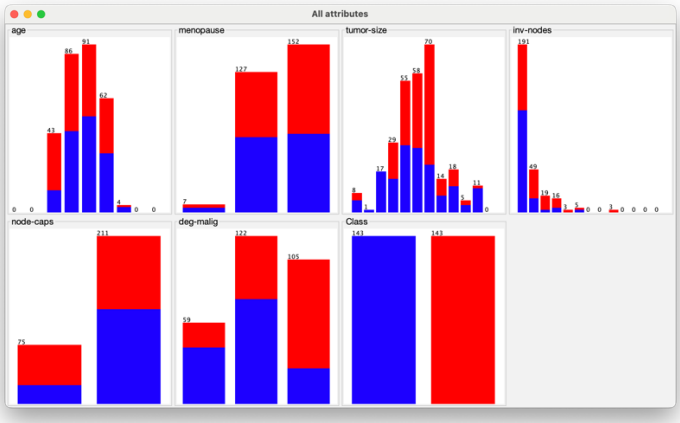
Esta eliminación busca reducir el ruido en los datos, mejorando la claridad del conjunto y evitando que atributos irrelevantes interfieran en la eficacia del modelo. Además, al trabajar con un menor número de atributos, se acelera el tiempo de entrenamiento y se optimiza el rendimiento general de los algoritmos de clasificación.





2.3 Remuestreo para Balancear Clases

El conjunto de datos original presentaba un desbalance significativo entre las clases *no-recurrence-events* y *recurrence-events*. Este desbalance puede inducir un sesgo en los clasificadores, favoreciendo la clase mayoritaria y dificultando la correcta identificación de la clase minoritaria. Para abordar este problema y mejorar el desempeño del modelo, se aplicó el filtro ***Resample*** con el parámetro biasToUniformClass = 1.0.



Este proceso de remuestreo es crucial para mejorar la capacidad del clasificador de reconocer ambas clases con precisión. Al balancear las clases, se optimiza el desempeño del modelo, lo que puede traducirse en una mejora de métricas clave como la precisión, especialmente al mejorar la capacidad para predecir correctamente las instancias de la clase menos representada.

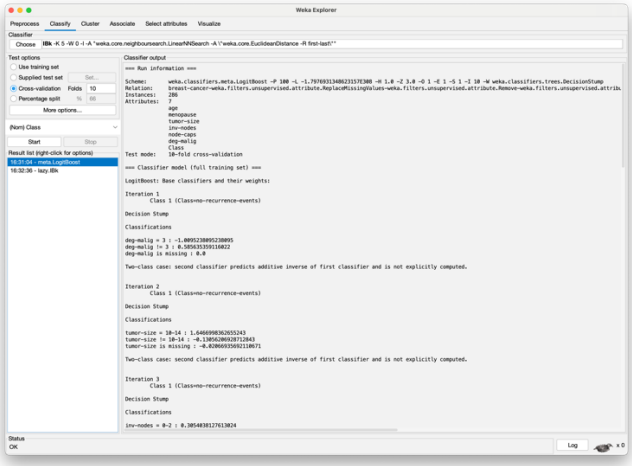
3.TÉCNICAS DE CLASIFICACIÓN

Para abordar la tarea de clasificación en el conjunto de datos, se seleccionaron dos algoritmos contrastantes: LogitBoost y k-Nearest Neighbors (k-NN). Estos algoritmos fueron evaluados utilizando validación cruzada con 10 particiones, lo que permite obtener una estimación más robusta del desempeño del modelo al promediar los resultados obtenidos en diferentes particiones del conjunto de datos.

3.1 LogitBoost

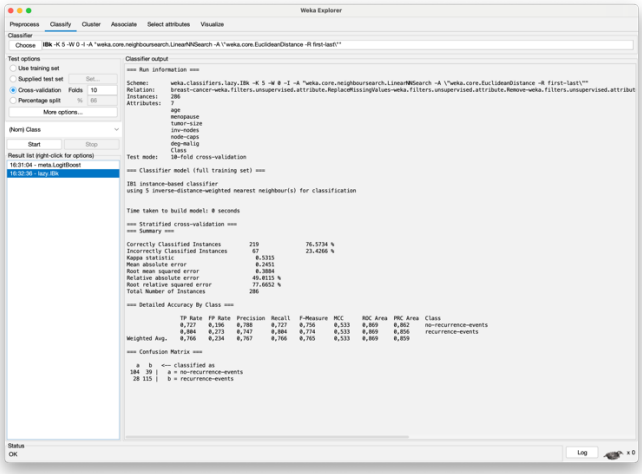
- **Descripción**: LogitBoost es un meta-clasificador que utiliza un conjunto de modelos base simples, como los Decision Stumps (árboles de decisión de un solo nivel), para formar un modelo final más robusto y preciso. En este enfoque, los modelos base son combinados de manera secuencial, donde cada modelo sucesivo intenta corregir los errores cometidos por los modelos anteriores.
  Este algoritmo se basa en la técnica de boosting, que combina modelos débiles (en este caso, los Decision Stumps) para crear un modelo final fuerte, que es capaz de capturar patrones complejos en los datos.
- **Configuración utilizada**:
  - Iteraciones: 10. Esto significa que el algoritmo crea 10 modelos base en su proceso de boosting.

- Clasificador base: Decision Stump. Un Decision Stump es un árbol de decisión muy simple, que solo tiene un nodo y toma decisiones basadas en una única característica. Este clasificador base es útil para crear un modelo robusto a partir de modelos sencillos.



3.2 k-Nearest Neighbors (k-NN)

- **Descripción**: k-Nearest Neighbors (k-NN) es un algoritmo basado en la clasificación por proximidad. Clasifica una nueva instancia según los k vecinos más cercanos a esa instancia en el espacio de características. La decisión de clasificación se toma en función de la mayoría de las clases entre los k vecinos más cercanos, utilizando una métrica de distancia, como la distancia Euclidiana.
- **Configuración utilizada**:
  - Número de vecinos: k=5. Esto significa que el algoritmo considerará los 5 vecinos más cercanos de cada instancia para determinar su clasificación.
  - Ponderación: Distancia inversa. Esto implica que los vecinos más cercanos tienen más peso en la decisión de clasificación. Los vecinos más alejados tienen un impacto menor en la clasificación, lo que ayuda a hacer la predicción más sensible a los datos cercanos.



4.RESULTADOS

4.1. Desempeño de LogitBoost

El modelo LogitBoost alcanzó una **exactitud global del 72.38%**, lo que indica un rendimiento aceptable pero limitado.

En la **matriz de confusión**, se observa un equilibrio relativo en las predicciones de ambas clases:

- 103 instancias correctamente clasificadas como *no-recurrence-events*.
- 104 instancias correctamente clasificadas como *recurrence-events*.

Sin embargo, también cometió 79 errores de clasificación (40 en la clase *a* y 39 en la clase *b*), afectando sus métricas de recall y precisión. Las métricas derivadas, como el **F1-score**, destacan:

- F1-score para *no-recurrence-events*: 0.72
- F1-score para *recurrence-events*: 0.72

La **área bajo la curva ROC (0.747)**, aunque moderada, muestra que LogitBoost tiene un desempeño aceptable al separar las dos clases, pero sufre debido al desbalance inicial del dataset.

Matriz de Confusión:

	Predicción: no-recurrence-events	Predicción: recurrence-events
Real: no-recurrence-events	103	40
Real: recurrence-events	39	104

Métricas por clase:

Clase	Precisión	Recall	ROC Area
no-recurrence-events	72.5%	72.0%	0.747
recurrence-events	72.2%	72.7%	0.747

4.2. Desempeño de k-NN (k=5)

El modelo k-NN obtuvo una **exactitud global del 76.57%**, superando a LogitBoost.

En la **matriz de confusión**, el modelo logró:

- 104 instancias correctamente clasificadas como *no-recurrence-events*.
- 115 instancias correctamente clasificadas como *recurrence-events*.

Reduciendo los errores totales a 67, k-NN mostró una mayor capacidad para capturar instancias de la clase minoritaria, como lo refleja su **recall del 80.4%** para *recurrence-events*. Las métricas derivadas también reflejan esta mejora:

- F1-score para *no-recurrence-events*: 0.76
- F1-score para *recurrence-events*: 0.77

Además, la **AUC-ROC de 0.869** demuestra una capacidad discriminativa significativamente mayor que la de LogitBoost. Esto sugiere que el modelo k-NN puede manejar mejor el balanceo previo de clases y aprovechar las relaciones locales en los datos.

Matriz de Confusión:

	Predicción: no-recurrence-events	Predicción: recurrence-events
Real: no-recurrence-events	104	39
Real: recurrence-events	28	115

Métricas por clase:

Clase	Precisión	Recall	ROC Area
no-recurrence-events	78.8%	72.7%	0.869
recurrence-events	74.7%	80.4%	0.869

5. ANÁLISIS Y CONCLUSIONES

Los resultados obtenidos demuestran que el modelo k-NN (k=5) ha superado a LogitBoost en varias métricas clave. Con una exactitud global de 76.57% frente al 72.38% de LogitBoost, k-NN mostró una mayor capacidad para clasificar correctamente tanto la clase mayoritaria como la minoritaria. Además, la AUC-ROC de k-NN (0.869) fue considerablemente superior a la de LogitBoost (0.747), lo que indica una mejor capacidad discriminativa, especialmente en un conjunto de datos desbalanceado.

Por otro lado, LogitBoost mostró un rendimiento moderado, con una AUC-ROC más baja y cometió más errores de clasificación, especialmente en la clase minoritaria (recurrence-events). Aunque LogitBoost es bueno para capturar relaciones complejas entre las variables, su rendimiento se vio afectado por el desbalance de clases, ya que tuvo dificultades para identificar correctamente la clase minoritaria.

En conclusión, k-NN (k=5) es más adecuado para este conjunto de datos, ya que maneja mejor el desbalance de clases y presenta un mejor desempeño general, convirtiéndolo en la opción más adecuada para este análisis.

Métrica	**LogitBoost**	**k-NN (k=5)**
Exactitud global	72.38%	76.57%
Kappa	0.4476	0.5315
AUC-ROC	0.747	0.869
Recall promedio	72.4%	76.6%
F1-score promedio	0.72	0.765