**SEDE:** ORELLANA

**FACULTAD:** INFORMÁTICA Y ELECTRÓNCA

**CARRERA:** INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

**MINERÍA DE DATOS**

**NIVEL:** SEXTO

**PARALELO:** A

**INFORME**

1. **DATOS GENERALES**

|  |  |
| --- | --- |
| **NOMBRES:** | **CÓDIGOS:** |
| Kerly Andi | 2836 |
| Diego Ramírez | 2874 |
| **FECHA DE REALIZACIÓN:** | **FECHA DE ENTREGA:** |
| 08/12/2023 | 11/12/2023 |

1. **TEMA:**

Algoritmos de aprendizaje supervisado

1. **DESARROLLO**

Con el fin de contribuir de manera precisa al artículo en desarrollo sobre la implementación de un modelo predictivo para prever la aparición de la enfermedad "escoba de bruja" en los cultivos de cacao mediante una aplicación híbrida que emplea arquitecturas limpias, se inicia la elaboración de este informe. En este documento, se presentará una explicación detallada y un análisis exhaustivo de los resultados obtenidos mediante diversos modelos, además de llevar a cabo una comparativa entre distintos algoritmos de Machine Learning para lograr un modelo predictivo altamente confiable. Estos algoritmos se agrupan en cuatro categorías específicas, diseñadas con el propósito fundamental de mitigar la complejidad y otros desafíos potenciales que podrían afectar la efectividad de este modelo predictivo. Posteriormente, se realizará la selección de la información más relevante para su inclusión en el artículo final.

Además, se generó un conjunto de datos consolidado que combinó la información recopilada de los sensores con los informes de las plantaciones de cacao obtenidos de manera manual. Posteriormente, se procedió a procesar estos datos, resultando en la creación de un archivo "dataset" en formato .csv, el cual se presenta a continuación:

Tabla

Descripción generada automáticamente

**Figura 1** Dataset

Así, se implementaron métodos con el propósito de evitar que este modelo se vea afectado por problemas relacionados con su complejidad o simplicidad, y estos métodos se categorizan en cuatro grupos distintos.

1. **Técnicas de la reducción de la dimensionalidad**

La reducción de la dimensionalidad es una técnica clave para simplificar conjuntos complejos de datos, donde el Análisis de Componentes Principales PCA, IPCA, KPCA que utiliza funciones de kernel como lineales, polinomiales y gaussianas para abordar datos no lineales, proyectándolos a espacios de mayor dimensión donde se vuelven linealmente separables, permitiendo un análisis más efectivo y conservando la información esencial para la toma de decisiones. En ese contexto en el dataset se va a identificar qué feature afecta a los modelos de Machine Learning.

Texto

Descripción generada automáticamente

**Figura 2** Algoritmo PCA, IPCA, KPCA

Los resultados obtenidos con los algoritmos mencionados anteriormente fueron los siguientes:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algoritmos** | | **Resultado** | **Análisis** | **Conclusión** |
| PCA | | 0.982089552238806 | El algoritmo PCA muestra el resultado con 7 variables utilizadas. | Al ejecutar el código y tras analizar los resultados se puede concluir que los algoritmos PCA, IPCA Y KPCA Linear dan los mismos resultados con un número igual de variables utilizadas. |
| IPCA | | 0.982089552238806 | El algoritmo IPCA muestra el mejor resultado con 7 variables utilizadas. |
| KPCA | LINEAR | 0.982089552238806 | El algoritmo con KPCA Linear muestra el mejor resultado con 7 variables utilizadas. |
| POLY | 0.9059701492537313 | El algoritmo KPCA polinomial muestra el mejor resultado con 6 variables utilizadas. |
| RBF | 0.8925373134328358 | El algoritmo con KPCA Gaussiano muestra el mejor resultado con 7 variables utilizadas. |

**Tabla 1** Datos normales, normalizados, discretizados

**Análisis**

Al culminar las pruebas y analizar los datos en la tabla presentada se puede concluir:

Los resultados obtenidos con los tres algoritmos mencionados anteriormente muestran que PCA, IPCA, KPCA Linear tienen el mismo resultado con 7 variables utilizadas, es decir, al final se puede elegir y trabajar con cualquiera de ellos para el entrenamiento de un modelo predictivo.

1. **Tendencia de regulación**

Continuando con las estrategias para evitar la complejidad y simplicidad excesiva en modelos predictivos, la tendencia de regulación se enfoca en penalizar aquellas características (features) que no contribuyen o restan información al modelo. Esta técnica abarca métodos como Lineal, Lasso, Ridge y ElasticNet.

Texto

Descripción generada automáticamente

**Figura 3** Regularización

Al aplicar esta técnica en el dataset muestra que feature tiene la variable de mayor peso para la predicción de la incidencia en las plantas de cacao.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algoritmos** | **Resultados** | **Variables** | **Análisis** | **Conclusión** |
| Score Lineal | [-0.01235473  0.42352019  -0.000967  0.00648492  -0.02038945  -0.07396781  0.05032552  -0.00536126] | Temperature  RH  Dew Point  Win Speed  Gust Speed  Wind Direction  Fruto  Severidad | Los coeficientes muestran la relación entre las variables y la incidencia. | Al usar los modelos Lineal, Lasso, Ridge, podemos notar que la variable ‘RH’ tiene el mayor peso, seguido por ‘Temperature’, siendo estas las más influyentes en la incidencia de la enfermedad. |
| Score Lasso | [ 0.  0.22430063  0.  -0.  0.  -0.  0.  0.] | Temperature  RH  Dew Point  Win Speed  Gust Speed  Wind Direction  Fruto  Severidad | Lasso reduce algunos coeficientes a cero, lo que indica una reducción de variables menos útiles. |
| Score Ridge | [-0.01232535  0.42327367  -0.00245916  0.00500751  -0.01908451  -0.0734277  0.04985505  -0.00532208] | Temperature  RH  Dew Point  Win Speed  Gust Speed  Wind Direction  Fruto  Severidad | Ridge mantiene todas las variables, reduce coeficientes, pero no los anula como Lasso. |
| Score ElasticNet | [ 0.  0.  0.  -0.  0.  0.  0.  0.] | Temperature  RH  Dew Point  Win Speed  Gust Speed  Wind Direction  Fruto  Severidad | ElasticNet también anula algunos coeficientes, manteniendo algunos similares a Lasso y Ridge. |

**Tabla 2** Comparativa de Algoritmos de Tendencia de regularización.

A continuación, se muestra los siguientes datos normales, normalizados y discretizados de cada modelo.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Algoritmos** | **Datos** | **Resultados** | **Análisis** | **Conclusión** | |
| Score Lineal | Normalizados | 0.8131069788672194 | Alta precisión del modelo lineal con datos normalizados. | Estos resultados destacan cómo la normalización y la discretización pueden impactar la precisión de los diferentes algoritmos de regresión lineal (Linear Regression, Lasso, Ridge, ElasticNet). En algunos casos, la discretización mejora la precisión, mientras que, en otros puede haber una leve disminución o un sobreajuste con los datos normalizados. |
| Discretizados | 0.7994962459957285 | Ligera disminución en la precisión con datos discretizados respecto a los normalizados |
| Score Lasso | Normalizados | 0.6346969423707713 | Precisión moderada del modelo Lasso con datos normalizados |
| Discretizados | 0.7188984336668203 | Mejora notable en la precisión del modelo Lasso con datos discretizados en comparación con los normalizados |
| Socre Ridge | Normalizados | 0.8131473202968755 | Precisión similar del modelo Ridge con datos normalizados en comparación con el modelo lineal |
| Discretizados | 0.7994911180118449 | Resultados muy cercanos a los obtenidos con datos normalizados en el modelo Ridge |
| Score ElasticNet | Normalizados | -0.005344818320704325 | Posible sobreajuste del modelo ElasticNet con datos normalizados |
| Discretizados | 0.27775052869071915 | Mejora significativa en la precisión del modelo ElasticNet con datos discretizados en comparación con normalizados |

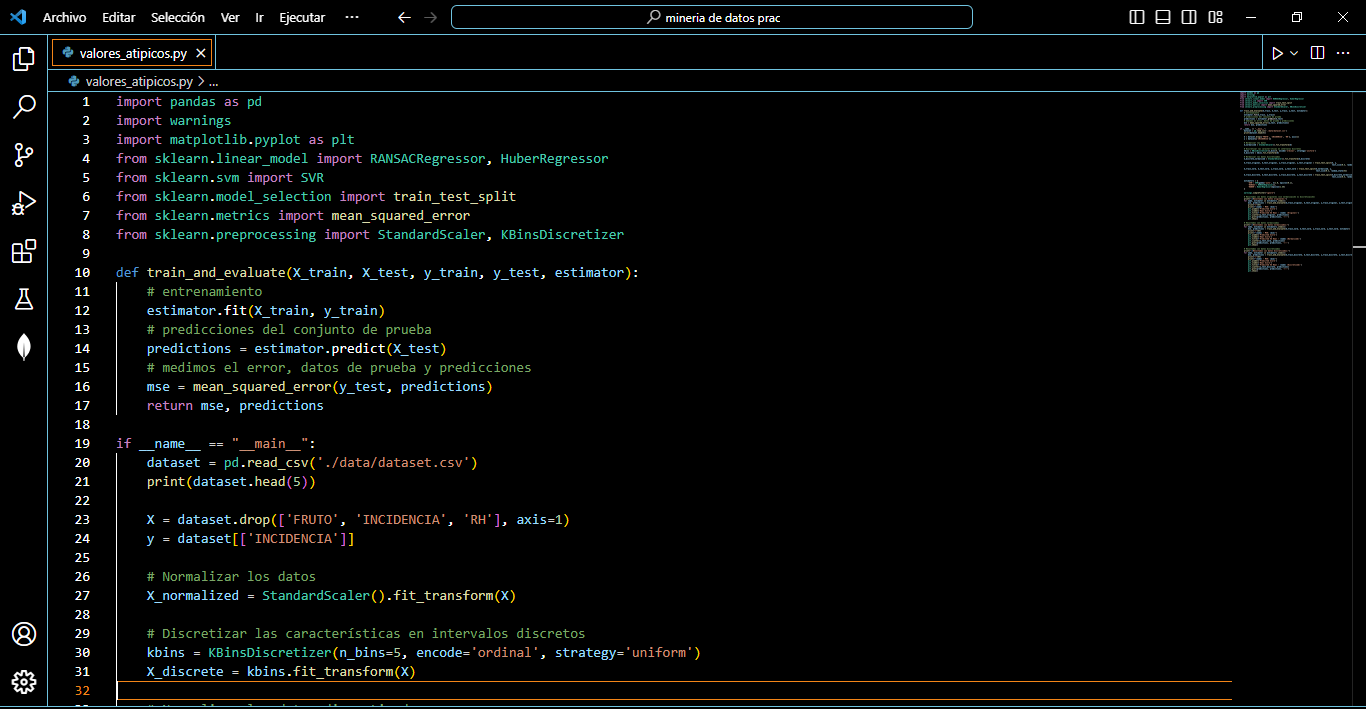
**Tabla 3** Datos normalizados discretizados

Al finalizar las pruebas y los análisis respectivos de los datos en la tabla anterior, se puede observar que estos resultados destacan cómo la normalización y la discretización pueden impactar la precisión de los diferentes algoritmos, en algunos casos, la discretización mejora la precisión, mientras que en otros puede haber una leve disminución o un sobreajuste con los datos normalizados.

Es importantes considerar el impacto del procesamiento en la precisión de los modelos al seleccionar el enfoque más adecuado para un conjunto de datos específicos.

1. **Valores atípicos con regresiones robustas**

Es crucial tener en cuenta la presencia de valores atípicos en nuestros dataset del proyecto, ya que pueden impactar la integridad de los modelos de regresión que estamos desarrollando. Para manejar esta situación, hemos optado por utilizar herramientas específicas de Scikit-learn, como SVR (Support Vector Regressor), RANSAC y Huber Regressor. SVR utiliza máquinas de vectores de soporte para definir un hiperplano que minimiza la región de respuestas de las muestras, lo cual resulta fundamental en nuestro análisis de regresión. RANSACRegressor emplea muestreo aleatorio para estimar parámetros de manera robusta, y HuberRegressor combina robustez con eficiencia. Estos modelos robustos son fundamentales para mejorar la resistencia de nuestros modelos a valores atípicos, permitiéndonos obtener estimaciones más confiables y precisas. A continuación, se muestra el modelo:



**Figura 4** Método de valores atípicos evaluados con datos normales, normalizados y discretizados**Análisis**

En este análisis se evaluaron los modelos (SVR), RANSAC y HUBER mediante pruebas con datos originales, datos normalizados y datos discretizados. Los resultados se mantuvieron consistentes en las tres configuraciones, y se presentan a continuación en forma general. Se empleó el Error Cuadrático Medio (MSE) como métrica de evaluación, destacando la variación en el rendimiento de los modelos ante diferentes manipulaciones de los datos.

**Resultados**

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Tipos de datos** | **MSE** | **Análisis** |
| **SVR** | Normales | 0.011049589945051435 | Presenta el mejor rendimiento con un MSE bajo, indicando una fuerte capacidad predictiva sin necesidad de normalización. |
| Normalizados | 0.018701276110738397 | La normalización apenas afecta su rendimiento, destacando la robustez de SVR ante variaciones en la escala de las características. |
| Discretizados | 0.029428991924605342 | Muestra un rendimiento aceptable, sugiriendo adaptabilidad a representaciones discretas. |
| **RANSAC** | Normales | 0.3417910447761194 | Indica una limitada capacidad de adaptación a cambios en los datos. |
| Normalizados | 0.3417910447761194 |
| Discretizados | 0.3417910447761194 |
| **HUBER** | Normales | 0.04778370476023218 | Puede ser considerado para un equilibrio entre robustez y tolerancia a atípicos. |
| Normalizados | 0.04766857354021391 | La invarianza a la escala se mantiene, mostrando consistencia en el rendimiento con datos originales y normalizados. |
| Discretizados | 0.05755596988170166 | Indica ciertas limitaciones al tratar con datos discretizados, aunque aún mantiene un rendimiento moderado. |

**Tabla 4**. Tabla comparativa de los algoritmos de métodos de valores atípicos con datos normales, normalizados y discretizados

**Conclusiones**

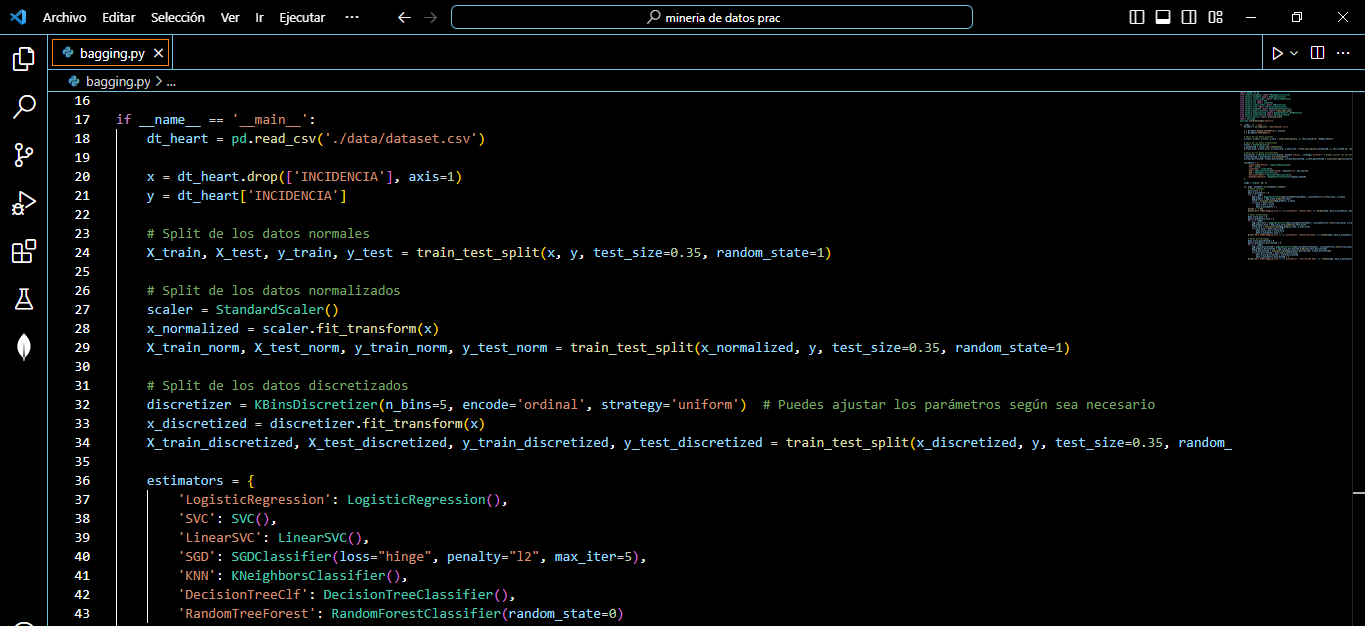
* SVR mantiene un rendimiento sólido en todas las configuraciones, destacando su robustez y capacidad para adaptarse a distintas manipulaciones de datos, lo que lo posiciona como la opción más confiable y versátil.
* RANSAC muestra consistencia en su rendimiento con un MSE constante en todas las configuraciones, indicando robustez, pero también sugiriendo limitaciones en la capacidad para mejorar la precisión en distintas representaciones de datos.
* HUBER presenta un rendimiento moderado, siendo una alternativa competitiva en términos de precisión y robustez, aunque ligeramente superado por SVR en este conjunto de datos específico.

1. **Métodos de ensamble**

Ahora nos enfocaremos en la aplicación de métodos de ensamble, los cuales involucran la combinación de diversos métodos con distintas configuraciones para lograr un consenso. Esta técnica se basa en la implementación de dos estrategias principales, las cuales son las siguientes:

**Estrategia de Bagging**

La estrategia de Bagging se centra en la reducción de la varianza de los clasificadores individuales mediante su combinación. A continuación, se detalla esta estrategia:



**Figura**  Estrategia de bagging con datos normales, normalizados y discretizados

**Análisis**

En la evaluación de estos algoritmos, estamos considerando la precisión como la métrica principal para evaluar la capacidad de clasificación. Estamos prestando especial atención al rendimiento en diversos contextos de datos, como datos normales, normalizados y discretizados, con el objetivo de comprender la adaptabilidad de cada algoritmo. Además, estamos analizando el número de estimadores en modelos de ensamble, como LogisticRegression, SVC, LinearSVC, SGD, KNN, DecisionTreeClf y RandomTreeForest, para evaluar la complejidad del modelo y su impacto en la precisión. Esperamos que un mayor número de estimadores ofrezca una capacidad superior para capturar patrones complejos, pero buscamos un equilibrio para evitar el sobreajuste. Este análisis preliminar establece las bases para evaluar de manera integral los algoritmos, considerando tanto su capacidad predictiva como su adaptabilidad a diversas condiciones de los datos.

**Resultados**

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmo** | **Mejor número de estimadores** | **Precisión** | **Análisis** |
| Logistic Regression | 2 | 1.0 | Excelente rendimiento con precisión perfecta. |
| SVC | 2 | 0.9795 | Adaptación sólida a la variabilidad de los datos. |
| LinearSVC | 2 | 1.0 | Capacidad excepcional para clasificar datos normales. |
| SGD | 6 | 1.0 | Rendimiento perfecto, eficaz en datos normales. |
| KNN | 16 | 0.9923 | Rendimiento excepcional, buena elección para este tipo de datos. |
| Decision Tree | 2 | 1.0 | Capacidad fuerte de clasificación con precisión perfecta. |
| Random Forest | 2 | 1.0 | Robustez destacada con precisión perfecta. |

**Tabla 5** resultados de los algoritmos con los datos normales

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmo** | **Mejor número de estimadores** | **Precisión** | **Análisis** |
| Logistic Regression | 6 | 0.9808 | Buen rendimiento, adaptación efectiva a la normalización. |
| SVC | 16 | 0.9744 | Sólido rendimiento, indicando adaptación a datos normalizados. |
| LinearSVC | 2 | 0.9885 | Buen desempeño, capacidad de clasificación eficaz con datos normalizados. |
| SGD | 18 | 0.9872 | Rendimiento sólido, adaptación efectiva a la normalización. |
| KNN | 14 | 0.9604 | Disminución en precisión, normalización afecta su rendimiento. |
| Decision Tree | 2 | 1.0 | Mantiene un rendimiento sólido incluso con datos normalizados. |
| Random Forest | 2 | 1.0 | Robustez destacada incluso con datos normalizados. |

**Tabla 6** Resultados de los algoritmos con los datos normalizados

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmo** | **Mejor número de estimadores** | **Precisión** | **Análisis** |
| Logistic Regression | 2 | 0.9706 | Precisión relativamente alta en datos discretizados. |
| SVC | 2 | 0.9706 | Mantiene solidez en la precisión con datos discretizados. |
| LinearSVC | 2 | 0.9706 | Capacidad constante de clasificación con datos discretizados. |
| SGD | 2 | 0.9706 | Rendimiento estable con datos discretizados. |
| KNN | 10 | 0.9604 | Disminución en precisión, discretización afecta su rendimiento. |
| Decision Tree | 12 | 0.9655 | Mantiene un rendimiento sólido con datos discretizados. |
| Random Forest | 18 | 0.9693 | Robustez destacada con datos discretizados. |

**Tabla 7** Resultados de los algoritmos con los datos discretizados

**Conclusiones**

Basándonos en los resultados proporcionados y considerando un equilibrio entre rendimiento y complejidad del modelo, el LinearSVC con datos normalizados y 2 estimadores podría considerarse como la mejor opción. Ofreciendonos un rendimiento sólido en diferentes condiciones y utiliza un número moderado de estimadores, lo que ayuda a evitar la complejidad excesiva del modelo.

**Estrategia Boosting**

La estrategia de boosting en aprendizaje automático mejora la precisión del modelo combinando múltiples modelos más débiles. Iterativamente, se enfoca en corregir los errores de los modelos anteriores al asignarles pesos según su desempeño. Este enfoque mejora significativamente la capacidad del modelo para capturar patrones complejos, logrando así un rendimiento más sólido. Ejemplos comunes de algoritmos de boosting incluyen AdaBoost y Gradient Boosting los cuales son los que vamos a usar para la evaluación de los datos.



**Figura**  Modelo Boosting

**Análisis**

Lo que se hace para implementar este método es que se utiliza los algoritmos de Gradient Boosting y AdaBoost para clasificar un conjunto de datos en tres configuraciones distintas: datos originales, datos normalizados y datos discretizados. Se realiza una búsqueda exhaustiva de la cantidad óptima de estimadores para cada algoritmo en cada configuración y se evalúa su desempeño en términos de precisión. Los resultados se almacenan y luego se imprimen, proporcionando la mejor configuración (número de estimadores y precisión) para cada algoritmo en cada tipo de datos. El código permite comparar cómo los algoritmos responden a diferentes formas de preprocesamiento, brindando insights sobre la robustez de los modelos ante variaciones en la naturaleza de los datos.

**Resultados**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Datos** | **Algoritmo** | **Mejor Precisión** | **Número de Estimadores** | **Análisis Breve** |
| Original | Gradient Boosting | 1.0 | 4 | Logra una precisión perfecta, indicando una capacidad excepcional para aprender patrones complejos en datos originales. |
| Original | AdaBoost | 1.0 | 2 | Alcanza una precisión perfecta con solo 2 estimadores, mostrando eficiencia en corregir errores y mejorar el rendimiento. |
| Normalizado | Gradient Boosting | 1.0 | 4 | Mantiene una precisión perfecta en datos normalizados, destacando su consistencia y adaptabilidad a diferentes formas de datos. |
| Normalizado | AdaBoost | 1.0 | 2 | Demuestra eficacia incluso en datos normalizados, logrando una precisión perfecta con solo 2 estimadores. |
| Discretizado | Gradient Boosting | 0.9706 | 4 | Mantiene una alta precisión (97.06%) en datos discretizados, evidenciando su capacidad para lidiar con datos de naturaleza discreta. |
| Discretizado | AdaBoost | 0.9706 | 2 | Logra una precisión del 97.06% en datos discretizados, mostrando consistencia en diferentes contextos de datos. |

**Tabla 8** Tabla Comparativa de los resultados del modelo boosting

**Conclusiones**

* Si se prioriza la Precisión Absoluta Gradient Boosting es ligeramente superior en datos discretizados, pero ambos son bastante similares.
* Si se Busca Eficiencia Computacional AdaBoost podría ser preferible, ya que logra precisión perfecta con menos estimadores en datos originales y normalizados.