Clasificación de estrellas enanas y gigantes usando aprendizaje automático

Roberto Narvaez Hernandez
Ingenieria Fisica Industrial
Instituto Tecnológico y de Estudios Superiores de Monterrey
Monterrey, Nuevo León
A01208129@itesm.mx

Jesús Eduardo De Alejandro Villarreal
Ingeniería en Innovación y desarrollo.
Instituto Tecnológico y de Estudios Superiores de Monterrey
Monterrey, Nuevo León
A00821785@itesm.mx

Luis Miguel Maawad Hinojosa
Ingeniería en Tecnologías Computacionales
Instituto Tecnológico y de Estudios Superiores de Monterrey
Monterrey, Nuevo León
A01364701@itesm.mx

William Obando Castellanos
Licenciatura en Administración y Estrategia de Negocios
Instituto Tecnológico y de Estudios Superiores de Monterrey
Morelia, Michoacán
A0106733@itesm.mx

Abstract

El análisis de cuerpos celestes requiere del manejo de grandes cantidades de datos para poder obtener información relevante de estos. Es por esto que, la astronomía puede beneficiarse en gran medida de la ciencia de datos y técnicas de aprendizaje automático. En atención a esta área de oportunidad, se creó una pequeña aplicación para la clasificación estelar, con base en las características de las estrellas; tales como su distancia de la tierra y su color, el cual es determinante para su clasificación en gigantes o enanas. Esto se realizó siguiendo la metodología de CRISP-DM, mediante la cual el resultado obtenido fue un modelo con una precisión alrededor del 89%, lo cual es prometedor, sin embargo dejar espacio para la mejora. Este programa pretende ser una primera aproximación al uso de ciencia de datos con aplicación en la astrofísica.

1 Introducción

El aprendizaje automático o machine learning (ML) se refiere a la aplicación de inteligencia artificial para el desarrollo de algoritmos dinámicos de toma de decisiones, basados en el reconocimiento de patrones de datos de manera automatizada (El Bouchefry & de Souza, 2020). Además, su implementación permite la generación de programas de computadora que mejoran automáticamente su ejecución por medio de la experiencia.

La posibilidad de la tecnología ML de llevar a cabo análisis de grandes cantidades de datos, dar una predicción con base en la futura inserción de información, así como de proveer una respuesta con una alta precisión, manifiesta el gran potencial de la aplicación de esta tecnología en distintas áreas; especialmente en la astronomía y geociencias, en las que la implementación de algoritmos ML prevé ser de gran impacto (El Bouchefry & de Souza, 2020).

En astronomía, la catalogación de estrellas se realiza con base en su masa y temperatura, para lo cual existen diferentes sistemas utilizados hasta hoy, entre ellos el “Harvard Spectral Classification System”, basado en la catalogación de las estrellas de acuerdo a su tamaño y color como indicador de temperatura, y el sistema de clasificación Yerkes o MKK, el cual es más preciso por incluir la variable de luminosidad.

De acuerdo con el Harvard Spectral Classification System, las categorías son: O (azul), B (azul - blanco), A (blanco), F (blanco - amarillo), G (amarillo), K (naranja) y M(rojo). Las letras representan un diferente tamaño, color y temperatura, yendo de más grande y más caliente a más pequeño y menos caliente. Por consiguiente, en el grupo O entran las estrellas más grandes y más calientes, mientras que, en la clase M se encuentran las más pequeñas y menos calientes. Dentro de esta clasificación, la adición de un número del 0-9 a la letra aumenta la especificidad en la

clasificación del cuerpo estelar, siendo el subtipo 0 lo más caliente y 9 lo más frío (Brennan, P, 2021).

A lo largo del tiempo, el proceso de clasificación estelar se realizó mediante la observación; actualmente, la catalogación de las estrellas se realiza a través de la Sloan Digital Sky Survey (SDSS), un mapa tridimensional del universo construido a partir de espectroscopía e imágenes ópticas - infrarrojas de campo amplio. No obstante, deficiencias en la SDSS como lo son el manejo de una cantidad excesiva de datos y la baja precisión de la clasificación estelar (Chao et al. 2020) abre una ventana de oportunidad hacia la implementación de tecnología de aprendizaje automático para mejorar el proceso de clasificación de las estrellas.

En el presente proyecto de investigación se propone implementar un sistema de catalogación automática de ML usando un modelo de redes neuronales.

La creación de esta herramienta, pretende hacer más eficiente el proceso de catalogación estelar en términos de tiempo y esfuerzo; así como aumentar la precisión y probabilidades de éxito, evitando al mismo tiempo, re-catalogaciones y un exceso de flujo de datos.

2 Conceptos previos

Para este proyecto se utilizaron 3 tipos de técnicas, técnicas para el preprocesamiento de datos, los modelos utilizados para clasificar y técnicas de validación, a continuación se explica detalladamente cada una de ellas.

2.1 Preprocesamiento de datos

Para poder tener un primer aproximado de cuales variables son más significativas y que tanta redundancia hay entre las variables se puede usar la matriz de correlación, en la cual se calcula la similitud coseno entre cada par de variables que se puede tener.

$$similarity(A, B) = \frac{A \cdot B}{||A|| \, x \, ||B||}$$

Ecuación 1. similitud.

Para ayudar a que los modelos pueden clasificar mejor los datos, estos se pueden normalizar de diferentes maneras para este proyecto se usó el Standard Scaler de SciKit learn el cual normaliza los datos según la siguiente ecuación: z = (x - u) / s donde u es el promedio de los datos de esa variable, s es la desviación estándar y x es el valor que se va a normalizar.

2.2 Modelos

Los modelos de clasificación utilizados para predecir la clase de la estrella se encuentran a continuación:

2.2.1 Regresión Lineal

El algoritmo de regresión lineal utilizado en este proyecto es el de la librería de Scikit Learn, este cuenta con un método de solución uno contra todos, el cual consiste en equipar cada clase contra todas las demás. Se implementaran todos los modelos de solucionadores, de manera que se elija el mejor según nuestro dataset. En la siguiente figura (ecuación #) podemos observar la función logística utilizada para calcular la probabilidad de este modelo.

$$\hat{p} = h_{\vec{\beta}}(\vec{x}) = \sigma(\vec{\beta}^T \cdot \vec{x}) = \frac{1}{1 + e^{-\vec{\beta}^T \cdot \vec{x}}}$$

Ecuación 2. Función Logarítmica.

2.2.2 K-Nearest Neighbors

Este algoritmo es implementado de la librería Scikit Learn, este es un algoritmo de votos, en el cual un número K de vecinos a un vector del dataset son seleccionados, de manera que estos vectores sean los más cercanos, una vez seleccionados se genera un voto según las clases de dichos vecinos y se clasifica según la clase que más se repita. A continuación se muestra la métrica de Minkowski, utilizada para medir la distancia entre vectores (ecuación #).

$$L_p(\vec{x}, \vec{y}) = ||\vec{x} - \vec{y}||_p = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

Ecuación 3 Métrica de Minkowski .

2.2.3 SVM

Los SVMs generan separaciones lineales dentro del espacio de coordenadas generado por los vectores de un dataset. Para este proyecto estaremos utilizando el método SVC de Scikit Learn, el cual hace uso de vectores de soporte para su clasificación.

2.2.4 LDA

LDA o Linear discriminant analysis es un método estándar para reducir la dimensionalidad de un problema de clasificación al proyectar los datos n-dimensionales en una sola línea. Esta línea se elige de tal manera que las diferentes clases estén bien separadas ej: maximiza la separación entre clases sobre las variaciones intraclase. (Williams, 1979)

2.2.5 Red Neuronal

Una red neuronal artificial (RNA) es un paradigma de procesamiento de información que se inspira en la forma en que los sistemas nerviosos biológicos, como el cerebro, procesan la información. El elemento clave de este paradigma es la estructura novedosa del sistema de procesamiento de información. Está compuesto por una gran cantidad de elementos de procesamiento (neuronas) altamente interconectados que trabajan al unísono para resolver problemas específicos. Las RNA, como las personas, aprenden con el ejemplo. Una RNA se configura para una aplicación específica, como el reconocimiento de patrones o la clasificación de datos, a través de un proceso de aprendizaje. El aprendizaje en sistemas biológicos implica ajustes en las conexiones sinápticas que existen entre las neuronas. Esto también se aplica a las RNAs. (Maind et al. 2014)

En la libreria Sckit-Learn se implementan algunos modelos basados en redes neuronales, como lo son:

1. Bernoulli Restricted Boltzmann Machine (RBM)
2. Multi-layer Perceptron classifier
3. Multi-layer Perceptron regressor

A su vez, las redes neuronales también se pueden clasificar según el tipo de aprendizaje, es decir: supervisado, no supervisado, híbrido y por refuerzo.

2.2.6 Multi-layer Perceptron classifier

El perceptrón multicapa consiste en un sistema de neuronas o nodos interconectados. Es un modelo que representa un mapeo no lineal entre un vector de entrada y un vector de salida. Los nodos están conectados por pesos y señales de salida que son una función de la suma de las entradas al nodo modificadas por una simple función de transferencia o activación no lineal. Es la superposición de muchas funciones de transferencia no lineales simples lo que permite al perceptrón multicapa aproximarse a funciones extremadamente no lineales. Si la función de transferencia fuera lineal, el perceptrón multicapa solo podría modelar funciones lineales. Debido a su derivada fácil de calcular, una función de transferencia comúnmente utilizada es la función logística. La salida de un nodo se escala por el peso de conexión y se alimenta para ser una entrada a los nodos en la siguiente capa de la red. Esto implica una dirección de procesamiento de la información, por lo que el perceptrón multicapa se conoce como red neuronal de alimentación hacia adelante (Feed forward). La arquitectura de un perceptrón multicapa es variable, pero en general consta de varias capas de neuronas. La capa de entrada no juega ningún papel computacional, sino que simplemente sirve para pasar el vector de entrada a la red. (Gardner, et al., 1998)

2.2.7 Ensambles

El objetivo de los métodos de ensamble es construir una colección o conjunto de clasificadores individuales que sean diversos y precisos. Eso permite obtener decisiones de clasificación altamente precisas votando las decisiones de los clasificadores individuales en el conjunto. Dos de las técnicas más populares para construir ensambles son la agregación bootstrap y la familia de algoritmos Adaboost (Dietterich, 2000)

Algunos de estos métodos de ensamble, también llamados meta-clasificadores se encuentran implementados en la librería SciKit learn. Entre ellos podemos encontrar:

1. RandomForestClassifier
   Es un meta estimador que se ajusta a una serie de clasificadores de árboles de decisión en varias submuestras del conjunto de datos y utiliza promedios para mejorar la precisión predictiva y controlar el sobreajuste. (scikit-learn.org, 2020)

2. ExtraTreesClassifier
   Es un meta estimador que se ajusta a una serie de árboles de decisión aleatorios (también conocidos como árboles extra) en varias submuestras del conjunto de datos y utiliza promedios para mejorar la precisión predictiva y controlar el sobreajuste. (scikit-learn.org, 2020)

3. AdaBoostClassifier
   Es un meta estimador que comienza ajustando un clasificador en el conjunto de datos original y luego ajusta copias adicionales del clasificador en el mismo conjunto de datos, pero donde los pesos de las instancias clasificadas incorrectamente se ajustan de manera que los clasificadores posteriores se enfocan más en casos difíciles. (scikit-learn.org, 2020)

4. GradientBoostingClassifier
   GB construye un modelo aditivo de manera progresiva por etapas; permite la optimización de funciones de pérdida diferenciables arbitraria. (scikit-learn.org, 2020)

5. HistGradientBoostingClassifier
   Durante el entrenamiento, el "cultivador" de árboles aprende en cada punto de división si las muestras con valores perdidos deben ir en el hijo izquierdo o derecho, según la ganancia potencial. Al predecir, las muestras con valores perdidos se asignan al hijo izquierdo o derecho en consecuencia. Si no se encontraron valores perdidos para una característica determinada durante el entrenamiento, las muestras con valores perdidos se asignan al hijo que tenga más muestras. (scikit-learn.org, 2020)

   Este meta clasificador está basado en otro meta clasificador desarrollado por microsoft llamado LightGBM

6. LightGBM

   LightGBM es un framework de mejora de gradientes que utiliza algoritmos de aprendizaje basados en árboles. Está diseñado para ser distribuido y eficiente. (microsoft, 2019)

2.2.8 Validación Cruzada

Tal como lo explica el Dr. en inteligencia artificial, Jason Brownlee (2020) en su sitio Machine Learning Mastery: La validación cruzada es un procedimiento de remuestreo que se utiliza para evaluar modelos de aprendizaje automático en una muestra de datos limitada.

Este procedimiento o método tiene un solo parámetro llamado k. Este se refiere al número de grupos en los que se dividirá una muestra de datos determinada. Como tal, el procedimiento a menudo se denomina validación cruzada de k veces. Cuando se elige un valor específico para k, se puede usar en lugar de k en la referencia al modelo, por ejemplo, k = 10 se convierte en una validación cruzada de 10 veces o "10-fold cross-validation" en inglés.

La naturaleza de este método es iterativo, eso quiere decir que el conjunto de datos se dividirá en k y de manera iterativa se utilizará ese bloque para hacer testing en los datos.

La validación cruzada se utiliza principalmente en el aprendizaje automático aplicado para estimar la habilidad de un modelo de aprendizaje automático en datos invisibles. Es decir, usar una muestra limitada para estimar cómo se espera que funcione el modelo en general cuando se usa para hacer predicciones sobre datos no usados durante el entrenamiento del modelo.

Es un método popular porque es simple de entender y porque generalmente da como resultado un método menos sesgado o menos optimista.

2.2.9 Métricas

La evaluación del rendimiento del modelo de aprendizaje automático resulta fundamental para cuantificar su capacidad de generalización y garantizar que este permite generar predicciones futuras que sean precisas y acertadas. Para lograr este objetivo, se requiere llevar a cabo el cálculo de métricas sobre un conjunto de datos de prueba; a continuación se enlistan las más comunes:

- Precisión: Es una métrica denominada como la más directa para evaluar el rendimiento de clasificación del modelo (ecuación 4). Determina el porcentaje de elementos clasificados de manera correcta, entre valores de cero y uno, cuyo valor, entre más alto sea, indica mayor precisión del modelo.

$$Precisión = \frac{TP}{TP + FP}$$

Ecuación 4. Precisión. TP: positivos totales; FP: falsos.

- Puntuación F1: La puntuación F1 corresponde a una métrica de valores de 0 a 1, que determina la media de la armonización entre la precisión y exhaustividad o recall, siendo mayor al acercarse al 1, y viceversa, indica menos precisión y exhaustividad en valores cercanos a cero (ecuación 5).

$$F1 = 2 * \frac{precisión * recall}{precisión + recall}$$

Ecuación 5.F1.

2.2.10 Matriz de confusión

Una matriz de confusión es una técnica para resumir el rendimiento de un algoritmo de clasificación.

La precisión de la clasificación por sí sola puede ser engañosa si tiene un número desigual de observaciones en cada clase o si tiene más de dos clases en su conjunto de datos. El cálculo de una matriz de confusión nos puede dar una mejor idea de lo que está haciendo bien el modelo de clasificación y de los tipos de errores que está cometiendo. (Brownlee, 2020b)

Estos errores pueden ser falsos negativos o falsos positivos. Al poder visualizar esta información en la matriz de confusión podemos ir ajustando nuestro modelo acorde a nuestras necesidades.

3 Metodología

Para el desarrollo del presente proyecto se empleó la metodología Cross Industry Standard Process for Data Mining (CRISP-DM) (Figura 1), cuyas fases se describen a continuación.

A

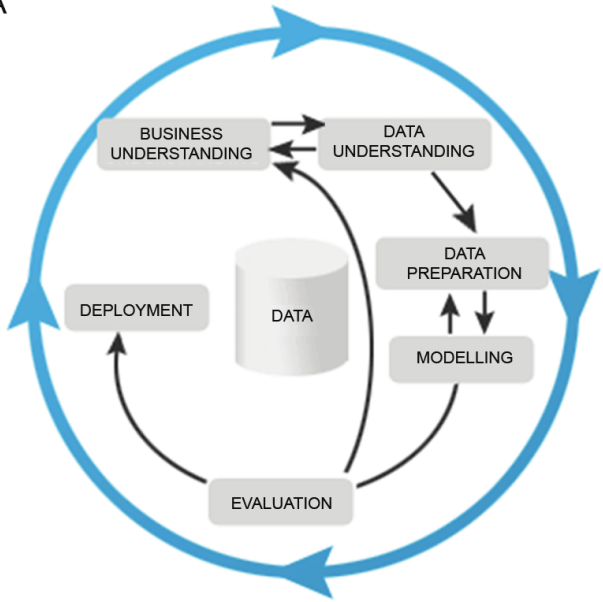

Figura 1. Diagrama de metodología CRISP-DM (Chapman et al. 2000).

3.1 Business understanding:

Este proyecto busca ayudar a los astrofísicos en la clasificación de estrellas, ya que esta labor suele ser muy tardada debido a que la cantidad de datos que se manejan suele ser muy grande, es por eso que se busca crear una herramienta que pueda resolver o facilitar dicho problema.

La primera pregunta que nos planteamos es cuáles serían las variables más útiles para clasificar estrellas? Otra pregunta también importante es cuál es el modelo que mejor podrá clasificar las estrellas?

3.2 Data understanding:

Nuestro proyecto se basó en el dataset: “Star Dataset: Stellar Classification” que se encuentra alojado en la comunidad online de científicos de datos llamada kaggle.com que a su vez es subsidiaria de Google.com.

El dataset a su vez se basa en el catálogo: “Hipparcos and Tycho” que tiene el nombre en código de “ESA 1997". Los datos de este catálogo fueron generados mediante observaciones del satélite Hipparcos de la agencia espacial europea.

A su vez, se pudieron acceder a estas observaciones mediante la herramienta online VizieR.

El dataset está formado por un total de 99,999 observaciones y 5 features, los cuales se enlistan a continuación:

- Vmag: Magnitud visual aparente de la estrella.
- Plx: Distancia entre la estrella y la tierra.
- e_Plx: Error estándar de Plx.
- B-V: índice de color.
- Sp Type:Tipo de espectro
- Target class: Si la estrella es enana (0) o gigante (1).

El conjunto de datos describe las características de los cuerpos estelares de acuerdo al sistema de clasificación MKK.

3.3 Data preparation:

Para preparar los datos se realizaron los siguientes pasos:

1. Lo primero que se hizo fue encontrar cuando datos faltantes se tenían, estos eran menos del 1% de los datos, por lo cual se optó por solo eliminar los renglones que contengan datos faltantes.
2. Posteriormente, eliminamos los renglones donde el error en la distancia entre la tierra y la estrella sea muy grande ya que estos datos no son confiables. Después de esto eliminamos la variable del set de datos ya que no representa utilidad para predecir la clasificación de la estrella.
3. Creamos una columna de magnitud absoluta (ecuación 6).
4. Normalmente la clasificación de estrellas tiene varias categorías pero para este problema solo nos enfocamos en las dos categorías principales que son estrellas gigantes y enanas para esto se tomó la columna de Sp Type y solo dejamos la clasificación de sí la estrella era gigante o enana.
5. Balanceamos los datos para tener la misma cantidad de datos de ambas categorías esto lo hicimos eliminando valores de la categoría que tenia mas datos para que ambas tuvieran el mismo tamaño.
6. Por último se visualizan para obtener una primera idea de que modelos puedan ser útiles.

$$MagA = Vmag + 5(log_{10} Plx + 1)$$

Ecuación 6: Cálculo de MagA

3.4 Modelling:

Se crearon dos grupos de features uno con los 4 features, y dado que la variable de Magnitud absoluta ya tomaba en cuenta las variables de magnitud relativa y distancia de la tierra a la estrella el otro grupo será con las variables de magnitud absoluta y B-V.

Para cada grupo de features se entrenaron 10 modelos diferentes usando validación cruzada, de estos se creó una gráfica de caja de cada uno de los modelos, estas se graficaron para posteriormente compararlos y elegir el mejor.

Todos los modelos que se entrenaron fueron de Scikit learn y fueron los siguientes:

- Regresion logistica (LR)
- Linear discriminant analysis (LDA)
- Máquina de Soporte vectorial (SVM)
- K vecinos cercanos con 5 vecinos (KNN)
- Random Forest Classifier (RFC)
- Extra Trees Classifier (ETC)
- AdaBoost Classifier (ABC)
- Gradient Boosting Classifier (GBC)
- Hist Gradient Boosting Classifier (HGBC)
- MLPClassifier, con una capa de 64 nodos y una de 32 (NN)

3.5 Evaluation:

Como se mencionó previamente se usó cross validation para evaluar los modelos y elegimos el modelo que tuviera el mayor promedio de accuracy y cuya gráfica de caja estuviera más acotada.

Posteriormente entrenamos el modelo elegido tomando 80% de datos de training y 20% de testing, después de entrenarlo se calculó su matriz de confusión, así como sus valores de Accuracy, Precision, Recall y F1 para así evaluar el desempeño del modelo final.

3.6 Deployment:

Durante esta fase se generó una aplicación que toma las variables “Vmag”, “Plx” y “B-V”, haciendo el cálculo de la última variable “MagA” según lo establecido previamente (ecuación 6).

De esta manera, y haciendo uso de él neural network y el standard scaler generados en la fase de modelación el programa clasifica esta estrella y le reporta el resultado al usuario.

4 Resultados

Al visualizar los datos en la figura 2, podemos observar que el arreglo de estos no muestra tener alguna distribución que sea fácil de separar linealmente, por lo tanto el uso de métodos lineales no promete ser la mejor opción, lo cual se pretende comprobar cuando se realice el entrenamiento de los modelos.

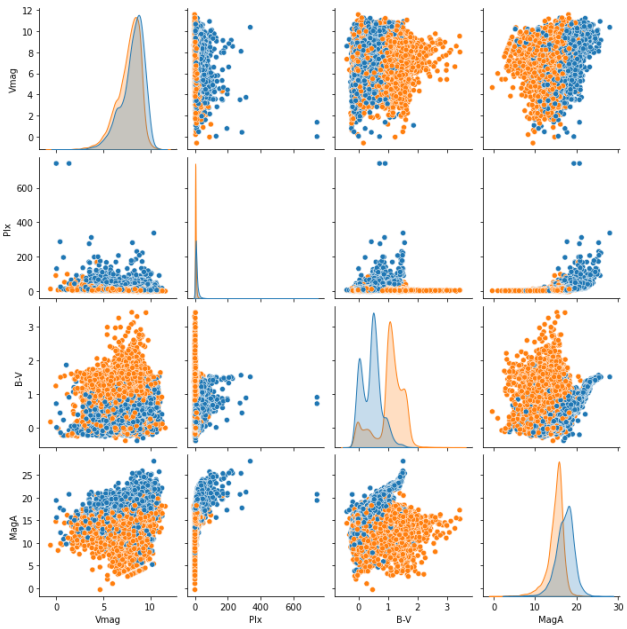

Figura 2. Distribución de los datos

Como se mencionó en la metodología, se entrenaron los modelos con dos sets de features diferentes, cuya gráfica de sus desempeños se observa en las figuras 3 y 4.

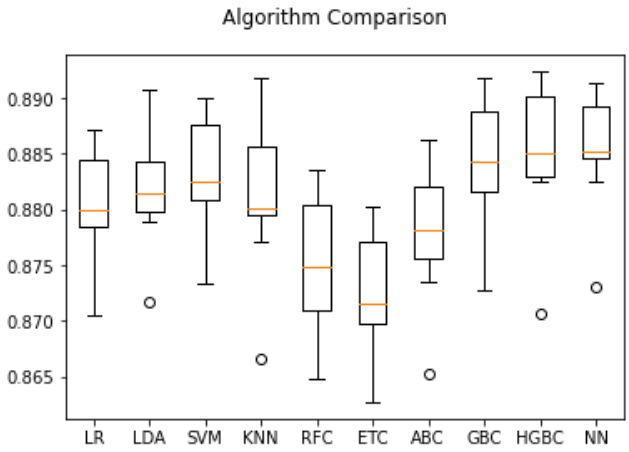

Figura 3. Modelos entrenados con todos los features

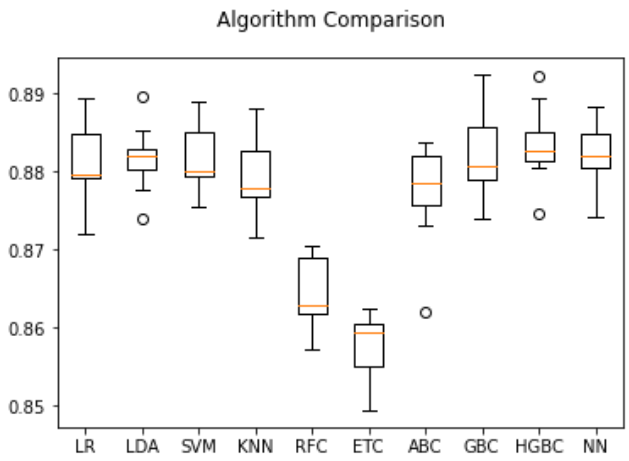

Figura 4. Modelos entrenados con 'MagA' y B-V''

A partir de la comparación de los algoritmos de los modelos entrenados (Figura 3 y 4), se observa que los métodos lineales tuvieron un mejor desempeño que varios métodos no lineales como lo son 12-NN y algunos métodos de ensamble, asimismo podemos ver que la diferencia entre los dos sets de modelos es muy pequeña, a pesar de que se puede ver una ligera mejora en el desempeño usando todos los features. Dentro de este set los modelos con mejor promedio de accuracy y con la menor variación son HGBD y la NN. Debido a que el HGBD aún es un método experimental, se optó por el uso de la NN para la creación de la aplicación final, asegurando de esta manera, disminuir la incidencia de error y aprovechar el espacio que la NN brinda para modificar varias variables con el fin de mejorar la precisión del modelo.

Una vez elegido el modelo de red neuronal como el más adecuado, se entrenó otra red neuronal usando 80% de los datos y dejando el 20% para testing. En la figura 5 es posible observar la matriz de confusión del modelo entrenado, a partir de la cual se obtuvo un accuracy de 0.8878, una precisión 0.8881, un recall de 0.8878 y un F1 de 0.8878. Los resultados de las métricas fueron sobre los datos de prueba, lo cual indica que el modelo tiene un buen desempeño no obstante, la visualización de los datos no muestra una distribución que sea fácil de clasificar y esto deriva en que la precisión del modelo no tenga la posibilidad de incrementar aún más.

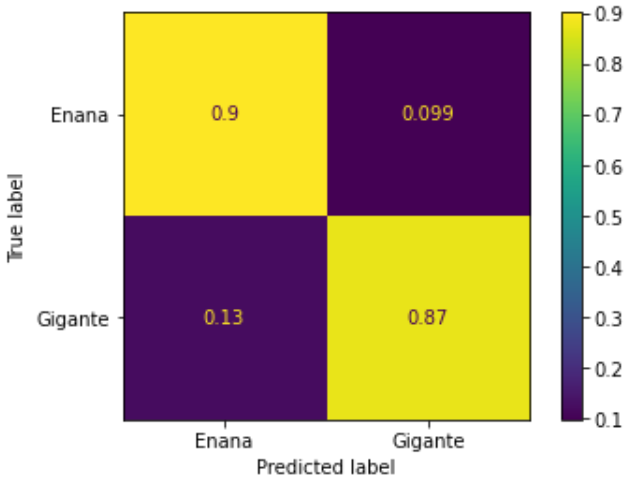

Figura 5. Matriz de confusión red neuronal

5 Conclusiones

En la visualización de los datos, la distribución aleatoria y poco clara de las clases mostró que sería difícil encontrar un modelo que correctamente clasificará dichos datos en su totalidad. En consecuencia, el modelo que tuvo mejores resultados tiene una precisión alrededor del 89%, lo cual es prometedor pero no excelente. Esta precisión se podría mejorar utilizando alguna técnica de análisis de variables que nos permita separar las clases de una manera que sea más fácil dividirlas; de igual forma, el resultado obtenido podría ser de gran utilidad para los astrofísicos, puesto que tiene la capacidad de brindar una primera aproximación sobresaliente sobre el tipo de estrella a clasificar.

Se planea en un futuro desarrollar una herramienta que permita a los astrofísicos llevar a cabo la clasificación de estrellas, a través de los datos que se provean de dichos cuerpos estelares, de manera que pueda ahorrarse tiempo al momento de la investigación y el rendimiento de su estudio sea mejor. Un prototipo de esta aplicación puede encontrarse en el código de este proyecto, el cual hace uso de nuestra neural network previamente entrenada, así como de un normalizador de datos, de manera que el usuario no tenga que normalizar la información de las estrellas de manera manual.

Referencias

[1] Astronomy and Astrophysics, 44(3), 345–355. https://doi.org/10.1016/j.chinastron.2020.08.005

[2] Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

[3] Brennan, P. (2021). What is an exoplanet? *NASA*. Retrieved from: https://exoplanets.nasa.gov/what-is-an-exoplanet/stars/#:~:text=Astronomers%20use%20these%20characteristics%20to,out%20of%20the%20main%20sequence.

[4] Brownlee, J. (2020, 2 agosto). A Gentle Introduction to k-fold Cross-Validation. Machine Learning Mastery. https://machinelearningmastery.com/k-fold-cross-validation/

[5] Brownlee, J. (2020b, agosto 15). What is a Confusion Matrix in Machine Learning. Machine Learning Mastery. https://machinelearningmastery.com/confusion-matrix-machine-learning/

[6] Chao, L., Wen-hui, Z., Ran, L., Jun-yi, W., & Ji-ming, L. (2020). Research on Star/Galaxy Classification Based on Stacking Ensemble Learning. Chinese

[7] Chapman, P.; Clinton, J.; Kerber, R.; Khabaz, T.; Reinartz, T.; Shearer, C.; Wirth, R. 2000. CRISP-DM 1.0: Step-by-step data mining guide. The CRISP-DM consortium. SPSS. Available at: http://www.spss.ch/upload/1107356429_CrispDM1.0.pdf

[8] Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine learning, 40(2), 139-157.

[9] El Bouchefry, K., & de Souza, R. S. (2020). Learning in Big Data: Introduction to Machine Learning. Knowledge Discovery in Big Data from Astronomy and Earth Observation, 225–249. https://doi.org/10.1016/b978-0-12-819154-5.00023-0

[10] Gardner, M. W., & Dorling, S. R. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. Atmospheric environment, 32(14-15), 2627-2636.

[11] Maind, S. B., & Wankar, P. (2014). Research paper on basic of artificial neural network. International Journal on Recent and Innovation Trends in Computing and Communication, 2(1), 96-100.

[12] Microsoft. (2019). microsoft/LightGBM. GitHub. https://github.com/Microsoft/LightGBM

[13] Star Dataset: Stellar Classification [Beginner]. (2020, 21 agosto). Kaggle. https://www.kaggle.com/vinesmsuic/star-categorization-giants-and-dwarfs

[14] scikit-learn.org. (2020). 1.11. Ensemble methods — Scikit-learn 0.24.2 documentation. https://scikit-learn.org/stable/modules/ensemble.html

[15] Williams, D. L., Stauffer, M. L., & Leung, K. C. (1979). A forester's look at the application of image manipulation techniques to multitemporal Landsat data. https://docs.lib.purdue.edu/lars_symp/301/