Tecnológico de Monterrey

Escuela de Ingeniería y Ciencias

Inteligencia Artificial Avanzada para la Ciencia de Datos

Análisis del desempeño de un modelo de Machine Learning

10/09/2023

Profesor:

Jesús Adrián Rodríguez Rocha

Autor:

Leonardo Ramírez Ramírez - A01351715

1. Introducción

Debido al gran desarrollo de las tecnologías de la información junto con su creciente implementación en tareas cotidianas dentro del quehacer humano, la dependencia de la correcta aplicación de las tecnologías se ha visto cada vez más palpable y necesaria para facilitar ciertas tareas. Con la avenida de esta revolución tecnológica, nació la necesidad de tener registros de información para analizarlos y asegurar estabilidad al tener un nivel de control sobre el futuro con las predicciones que pueden inferirse a través de los datos. Así fue como el Big Data transformó el paradigma contemporáneo actual y con ello surgió la necesidad de asignar normas y reglas a las bases de datos con el fin de asegurar un correcto uso.

Según la compañía u organismo nacional o internacional puede existir ciertas reglas para respetar el uso y privacidad de los datos contenidos en ciertas bases de datos. Por ello, surge la obligación como ingenieros de datos y científicos de datos de buscar la normatividad asociada al dataset en uso y respetar los lineamientos estipulados por los derechos de autor, reglas de uso, etc.

2. Modelo de Machine Learning

La razón principal por la que se eligió utilizar los modelos de *Regresión Logística Multiclase* y *Random Forest* fue por la selección del dataset pues, desde un inicio, se tuvo pensado en implementar el modelo de ML para problemas de astrofísica o relacionados dentro del quehacer en astronomía. Por ello, obtuvimos el dataset desde un repositorio de GitHub de Kaggle el cual puede econtrarse como Dataset.

2.1. Sobre el database

La base de datos contiene varios features tanto numéricos como categóricos de temperatura, luminosidad, color, clasificación espectral y tipo de estrella.
La tarea es clasificar el tipo de estrella según sus demás parámetros numéricos. Los tiplos disponibles de estrella son:

Valor	**Tipo de estrella**
0	Brown Dwarf
1	Red Dwarf
2	White Dwarf
3	Main Sequence
4	Supergiant
5	Hypergiant

2.2. Separación de datos en Train/Test

La separación de la base de datos en conjuntos de entrenamiento y prueba es una práctica común e importante al implementar un modelo de ML. Separar los datos es posible entrar al modelo con el conjunto de entrenamiento para que éste aprenda las relaciones subyacentes entre las variables, identifique los patrones y extraiga la información relevante de la distribución de los datos para finalmente verificar su rendimiento en el conjunto de prueba a fin de estimar cómo se comportará el modelo al implementarse con datos que aún no ha visto. De forma general este es parte del procedimiento necesario para evaluar el desempeño y la generalización del modelo [1].

En el dataset utilizado se seccionaron los datos en conjunto de entrenamiento y conjunto de prueba utilizando la función *train_test_split* parte del módulo de ML en python, **Sci-kit learn**. La proporción seleccionada fue 80 % del total de datos en el conjunto de entrenamiento y el 20 % restante en el conjunto de prueba. Gracias a esta proporción fue posible entrenar al modelo con una suficiente cantidad de datos para después evaluar su desempeño en el conjunto de prueba.

2.3. Métrica de evaluación de rendimiento

Para medir qué tan bien el modelo se desempeña haciendo predicciones a partir del conjunto de prueba es una manera de validar si existe subajuste o sobreajuste durante el proceso de entrenamiento. El sobreajuste es posible detectarlo si el rendimiento del modelo en el conjunto de prueba es significativamente peor que en el conjunto de prueba [1].
Para lograr este primer acercamiento a la evaluación del modelo de ML sobre datos nuevos se utilizan métricas proporcionadas por **Sci-kit learn** como *accuracy*, *recall*, *métrica F1*, *matriz de confusión*, entre otras más. Al tratarse de un modelo de calificación multiclase, la evaluación y validación del modelo se realizó según su precisión (*accuracy*).

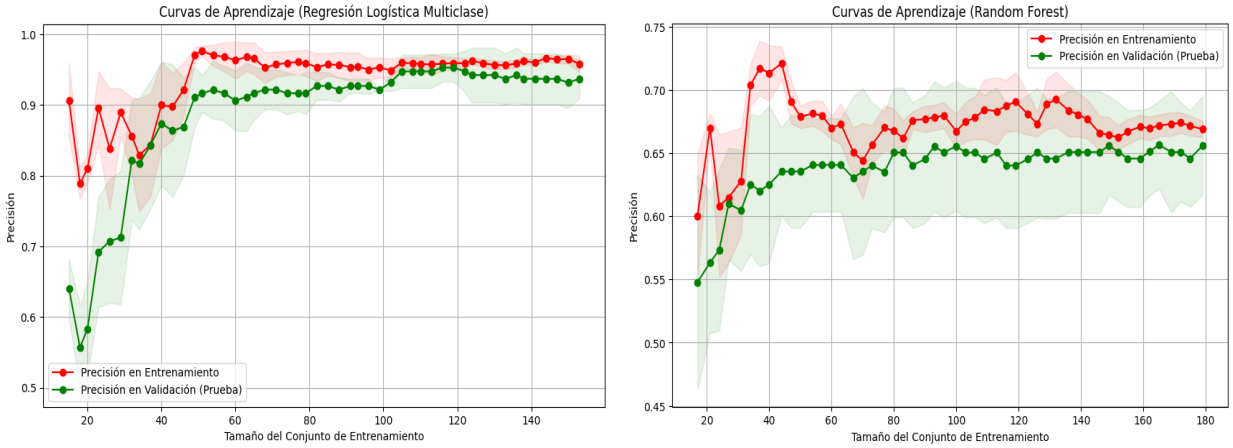
El primer intento de aplicación de los modelos arrojó resultados de precisión muy buenos pues ambos tuvieron un *accuracy* de aproximadamente $90 - 95\,\%$. Por ello, se tuvo que recurrir a un caso especial en el que se empeoró el modelo con tal de obtener puntajes de precisión más bajos. Esto se realizó cambiando los hiperparámetros de ambos modelos hasta que se obtuvo:

Modelo	**Precisión**
Regresión Logística Multiclase	96 %
Random Forest	60.4 %

2.4. Análisis del bias y varianza: Diagnóstico de subajuste y sobreajuste

La curva de aprendizaje es una herramienta bastante útil para identificar visualmente cómo mejora el modelo de ML a medida que se le proporcionan más datos. Se utilizó la función, integrada en **Sci-kit learn**, de *learning_curve* la cual determina un entrenamiento y puntajes de prueba a través

de aplicar validación cruzada. Esto con la finalidad de estimar como el modelo generaliza a partir de conjuntos de entrenamiento de diferente tamaño. La función estima el rendimiento de manera robusta aplicando la validación cruzada *Stratified K-fold validation* y comparando la precisión como la métrico de evaluación en cada *fold*.
En la figura 1 se muestra la curva de aprendizaje obtenida para ambos modelos.



(a) Curva de aprendizaje de Regresión Logística (b) Curva de aprendizaje de Random Forest

Figura 1: Comparación entre la precisión en entrenamiento y prueba de ambos modelos.

Es posible observar en la curva de aprendizaje del modelo de *Regresión Logística* cómo, a medida que aumenta el tamaño de los conjuntos de entrenamiento, el modelo presenta una creciente mejora en la precisión de validación llegando a estabilizarse respecto a la precisión del conjunto de entrenamiento. Además, se aprecia cómo tiene un comportamiento más estable y con mejores valores de precisión en ambos conjuntos donde en conjunto de prueba alcanza una precisión por encima del 90 %. Esto se explica debido al alto grado de precisión que se obtuvo desde un primer inicio debido enteramente a la naturaleza y simplicidad de la base de datos.
Por otro lado, la curva de validación del modelo de *Random Forest* no presenta tan buenos valores de precisión en ambos conjuntos y no llega a estabilizarse a medida que aumenta el tamaño del conjunto de entrenamiento. Además, se observa que la variabilidad (laa partes sombreadas de cada curva) es mucho mayor en comparación con la curva de aprendizaje obtenida para el modelo de regresión. Esto, a la par de visualizar un comportamiento un tanto más errático, se explica debido a la baja precisión obtenida por nuestro modelo desde un inicio como se mostró en la tabla anterior.

- **¿Qué nos dice acerca del subajuste y sobreajuste?**

Las curvas de aprendizaje revelan información acerca de si el modelo está presentando algún problema de sobreajuste o subajuste. Por ejemplo, es posible observar en la figura 1.(a) un problema

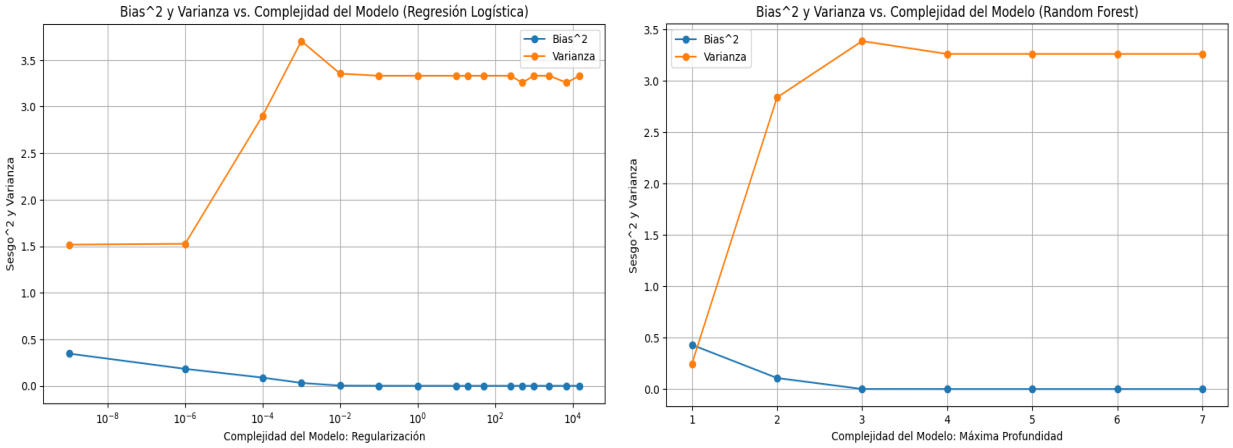
de sobreajuste (**overfitting**) al existir una diferencia significativa entre la precisión del modelo en el entrenamiento contra el rendimiento de las predicciones en el conjunto de validación. A medida que el tamaño del conjunto de entrenamiento aumenta, el **overfitting** disminuye al aumentar la precisión del modelo en el conjunto de validación y estabilizarse muy cerca de la precisión de entrenamiento. Alrededor de un tamaño de entrenamiento de 100 elementos en adelante, la diferencia entre ambas curvas deja de ser significativa llegando a superar el problema de sobreajuste y estabilizándose en un rango de muy buenas predicciones.

Por otro lado, en la figura 1.(b) se tiene posiblemente un problema de subajuste (**underfitting**) debido a que tanto la precisión del modelo en el conjunto de entrenamiento como en el conjunto de prueba son muy bajas. A pesar de que la curva de precisión de entrenamiento está por encima de la curva de prueba a lo largo de la gráfica, el hecho de presentar valores tan bajos de precisión para ambos casos está posiblemente relacionado a que el modelo no es capaz de extraer correctamente los patrones y relaciones de las variables de la base de datos. Esto es un indicativo de que los hiperparámetros seleccionados para el modelo de *Random Forest* no fueron los óptimos para detectar de manera óptima los patrones subyacentes en los datos.

- **¿Qué sucede con el bias y la varianza del modelo?**

En Machine Learning, existe una relación importante entre el **bias** (o sesgo), y varianza de un modelo. El **bias** es una medida para cuantificar la simplificación excesiva del modelo para ajustar los valores reales de entrenamiento. Generalmente se considera a un modelo con alto sesgo como un modelo simple que no es capaz de extraer la complejidad de los patrones en la base de datos, teniendo un problema de underfitting.
Por otro lado, la **varianza** es una medida que cuantifica la sensibilidad del modelo a variaciones en los datos de entrenamiento. Un modelo con alta varianza se asocia a un problema de overfitting pues el modelo se ajusta demasiado bien a los patrones de lo datos de entrenamiento, incluso llegando a capturar el ruido ocasionando que el modelo tenga un deficiente rendimiento cuando se enfrenta a datos de prueba, presentando muy mala generalización [1].

En la figura 2 se muestra visualmente el comportamiento de las gráficas de sesgo y varianza para cada uno de los modelos. Dicho comportamiento se grafica con respecto al incremento en la complejidad del modelo. Para el modelo de regresión la complejidad va aumentando según la variación de su parámetro de regularización C el cuál controla la restricción en el aprendizaje de los datos (para valores pequeños, mayor es la regularización) [2].
Por otro lado, para el modelo de random forest se utilizó la profundidad máxima como hiperparámetro característico de complejidad del modelo.



(a) Bias y Varianza de Regresión Logística (b) Bias y Varianza de Random Forest

Figura 2: Comportamiento del bias y varianza en ambos modelos

En la figura 2.(a) observamos un caso particular pues, como se vió anteriormente, el modelo de regresión logística es un modelo que presenta alta varianza desde un principio (incluso con los peores hiperparámetros encontrados para bajar su rendimiento). Esto indica que el modelo presenta un gran problema de overfitting pues se observa en la gráfica cómo aumenta la varianza y gradualmente disminuye el sesgo a medida que la complejidad del modelo aumenta. Esto indica que debe realizarse una modificación en los hiperparámetros del modelo para disminuir su complejidad, pues como se observa, la varianza va incrementando cada vez más conforme lo hace el parámetro de regularización (e utilizó una escala logarítmica en el parámetro de C debido a que son valores demasiado cercanos a 0).
En la figura 2.(b), por otro lado, se observa el comportamiento del sesgo y varianza del modelo de random forest. En ella se aprecia cómo la varianza va aumentando rápidamente conforme lo hace la profundidad máxima del modelo. Sin embargo, aquí se aprecia una intersección en el aumento del sesgo y la varianza. Esto indica que desde el comienzo el modelo presentó un caso ligero de underfitting y luego pasó a tener un comportamiento de overfitting rápidamente.

2.5. Refinamiento del modelo

Para refinar el rendimiento general de los modelos se procedió a utilizar la función *GridSearchCV* parte del módulo de **Sci-kit learn**. Esta herramienta se utilizó como método de búsqueda y selección de la mejor combinación de hiperparámetros mediante una búsqueda exhaustiva de un conjunto de hiperparámetros previamente definidos. Esto se realizó utilizando una búsqueda de cuadrícula entre las combinaciones mientras se evaluaba el rendimiento del modelo con dichos hiperparámetros utilizando validación cruzada.

La mejor combinación de hiperparámetros para ambos modelos se muestra a continuación:

Regresión Logística	
Hiperparámetros	**Mejor modelo**
Multi_class	auto
Random_state	42
C	100
Solver	lbfgs
Penalty	l2
Max_iter	1000

LogisticRegression(multi_class $=' auto', C = 100$, penalty $=' l2'$, solver $=' lbfgs'$, max_iter $=$ 1500, random_state $= 42$)

Random Forest	
Hiperparámetros	**Mejor modelo**
Max_depth	None
Min_samples_leaf	1
Min_samples_split	2
n_estimators	1
Criterion	Entropy
Random_state	42

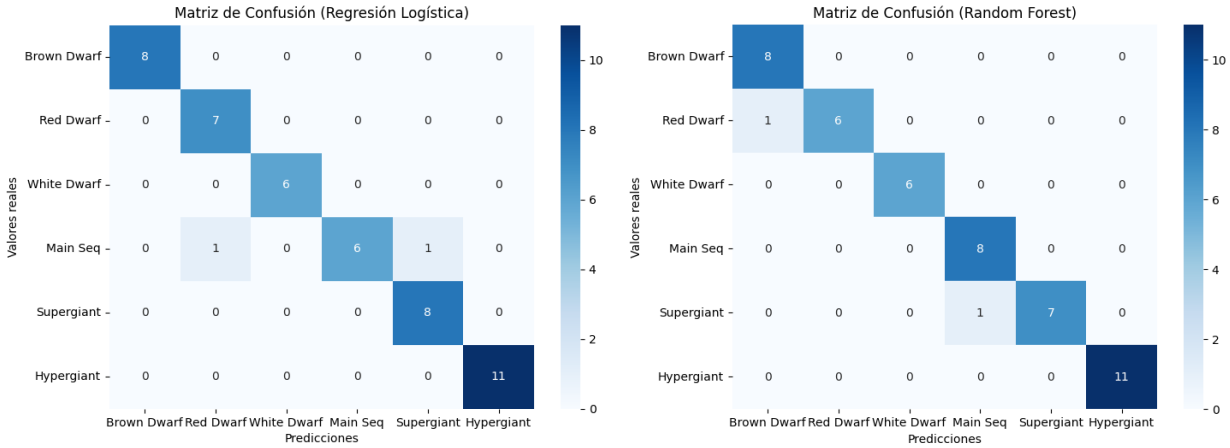
RanfomForestClassifier(n_estimators $= 1$, max_depth $= None$, criterion $='$ $entropy'$, random_state=42, min_samples_leaf $= 1$, min_samples_split $= 2$)

La métricas de evaluación de rendimiento utilizadas para demostrar que en efecto encontramos la combinación óptima de hiperparámetros fue la precisión del modelo y su matriz de confusión. La precisión obtenida para cada modelo fue casi perfecta, siendo ambas más del 95 %.

Modelo	**Precisión**
Regresión Logística Multiclase	98 %
Random Forest	95.83 %

A partir de la precisión ya podemos inferir que, en efecto, hubo una buena mejoría de los modelos. Para reafirmarlo, en la figura 3 se muestran las matrices de confusión para cada uno de los modelos. En ellas se aprecia que ambos modelos clasifican correctamente los tipos de estrellas teniendo unos menores problemas clasificando algunas otras. Sin embargo, de manera general se aprecia que la gran mayoría son clasificadas correctamente según el dataset de entrenamiento.
Analizando, además, los valores de los hiperparámetros encontrados encontramos que concuerdan

para mejorar el rendimiento según el comportamiento de los primeros modelos. Un ejemplo claro es la simplificación de la complejidad del random forest según su profundidad máxima pues, como vimos anteriormente, a bajas profundidas ya se lograba tener problemas de overfitting. Por ello, *GridSearchCV* encontró un valor nulo para la profundización pero lo quilibró con otros hiperparámetros.



(a) Matriz de confusión de Regresión Logística (b) Matriz de confusión de Random Forest

Figura 3: Matriz de confusión para cada modelo

3. Conclusión

Al implementar un modelo de aprendizaje automático (ML) para abordar problemas de clasificación o regresión, resulta fundamental evaluar el rendimiento del modelo mediante métricas de validación apropiadas, tales como la precisión, la matriz de confusión o la curva de aprendizaje, entre otras. Esta evaluación reviste una importancia crucial a fin de determinar la capacidad de generalización del modelo y su desempeño en datos no observados, es decir, en datos de prueba.

Adicionalmente, es esencial llevar un registro detallado del sesgo y la varianza inherentes al algoritmo de ML utilizado. Este registro permite identificar posibles problemas de underfitting (subajuste) y overfitting (sobreajuste), los cuales son desafíos comunes en la construcción de modelos. Aunque las curvas de aprendizaje proporcionan información valiosa sobre estos problemas, realizar un diagnóstico preciso del sesgo y la varianza es el enfoque necesario para confirmar su existencia y, de manera aún más importante, determinar estrategias efectivas para abordarlos de manera adecuada. En la parte del refinamiento del modelo fue esencial utilizar una función de búsqueda y comparación de combinación óptima de parámetros. Gracias al aprovechamiento de las funciones integradas dentro del módulo de **Sci-kit learn** es que fue posible realizar el presente trabajo.

Referencias

[1] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

[2] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, (2011). scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.