Aprendiendo Machine Learning con Python

Eliana Bonalde

Universidad Industrial de Santander
Maestría en Matemática Aplicada
Modelado matemático

12 de junio de 2021

Índice

1. Introducción

El Machine Learning, a través de métodos estadísticos y algoritmos, habilita al computador a aprender por medio de la identificación de patrones, descubriendo información clave dentro de un conjunto de datos, para luego hacer predicciones o clasificaciones. Actualmente, con tanta información que podemos intercambiar a través de la web, las técnicas de Machine Learning ayudan a relacionar las distintas variables en un proceso, reduciendo el margen de error en la estimación de resultados. Es interesante entender qué es y cómo se usa, así como también reconocer los problemas que pueden ser resueltos por esta metodología. Este proyecto busca compartir una experiencia pedagógica sobre los primeros pasos de Machine Learning, está dividido en tres secciones: preparación de los datos, creación de modelos y validación y optimización de los modelos.

2. Machine Learning

El Machine Learning, o Aprendizaje Automático, es una forma de la Inteligencia Artificial que permite a un sistema aprender de los datos en lugar de aprender mediante la programación explícita [1], a través de métodos estadísticos y algoritmos, habilita al computador a aprender por medio de la identificación de patrones, descubriendo información clave dentro de un conjunto de datos, para luego hacer predicciones o clasificaciones.

Existen tres tipos de Machine Learning: supervizado, no supervizado y reforzado.

1. Supervizado: Conocemos tanto las variables de entrada (características) como la variable de salida (etiqueta o variable objetivo). Principalmente, la máquina recibe datos que se caracterizan por las variables X, etiquetados o clasificados en una variable y, y de esta manera la máquina aprenda a predecir el valor y de acuerdo a las características X que se le asignen.

2. No supervisado: En éste, los datos de entrenamiento no incluyen las etiquetas, el algoritmo intentará clasificar o descifrar la información por sí solo mediante un proceso iterativo, analizando los datos sin intervención humana.

3. Reforzado: Es un modelo de aprendizaje conductual. El algoritmo recibe retroalimentación del análisis de datos, conduciendo el usuario hacia el mejor resultado a medida que obtiene premios o penalidades, donde una secuencia de decisiones exitosas conduce al fortalecimiento del proceso, resolviendo el problema de manera más efectiva[1].

Las etapas en un proceso de Machine Learning la podemos dividir de la siguiente manera:

- Recopilar y preparar un conjunto de datos históricos de valor.
- Crear un modelo con los datos (entrenar un algoritmo).
- Evaluar.
- Configuración de parámetros.
- Hacer predicciones.
- Interpretar del modelo.

3. Metodología

Las etapas de Machine Learning se presentan en tres secciones: preparación de datos, modelos y evaluación y optimización de modelos. Cada una de estas secciones fueron trabajadas con el lenguaje de programación Python en Jupyter Notebook.

3.1. Preparación de los datos

El primer paso, y el más importante del diseño del modelo, es entender qué información contiene cada variable y detectar posibles errores, ya que la calidad y cantidad de información que se consiga impactará directamente en la precisión del modelo.

Se utilizó la librería Pandas de Python [18] en toda la sección, y los pasos seguidos fueron los siguientes:

1. Se importa un archivo en formato .csv donde se encuentra el conjunto de datos 1 [5] y se convierte en un DataFrame de Pandas para su mejor visualización. En él se observa información sobre dos tipos de variables:

   a) Las variables predictoras, también llamadas características.
   - id: Identificación
   - gender: Género
   - age: Edad
   - hypertension:"0" si el paciente no tiene hipertensión. "1" si el paciente tiene hipertensión.
   - heart_disease: "0" si el paciente no tiene ninguna enfermedad cardíaca. "1" si el paciente tiene una enfermedad cardíaca.
   - Residence_type: Tipo de residencia.
   - avg_glucose_level: Nivel medio de glucosa en la sangre.
   - bmi: Índice de masa corporal.
   - smoking_status: "anteriormente fumado", "nunca fumado", "fuma" o "Desconocido".

   b) La variable objetivo, la cual representa lo que queremos predecir con el modelo de Machine Learning.
   - stroke: "1" si el paciente tuvo un accidente cerebrovascular, "0" si no.

2. Se observa si existen datos no nulos y, en caso de haberlos, esto se puede resolver de las siguientes formas: eliminarlos completamente, es decir, eliminar las filas donde falte información; o sustituirlos por algún valor, por ejemplo, la media. Se recomienda eliminarlos, ya que no se estaría alterando la información.

3. Se eliminan las columnas que no contienen información relevante para el modelo.

4. Hay que asegurarse que el tipo de datos sea "numérico", para que posteriormente no haya problemas a la hora de ajustar los modelos de Machine Learning.

5. Se dividen los datos en dos subconjuntos: de entrenamiento y de prueba. Lo anterior es para evitar el sobreajuste que tenemos cuando se evalúa y se predice el rendimiento del modelo sobre los mismos datos de entrenamiento, ya que éste está familiarizados con estos últimos, siendo muy difícil obtener idea del comportamiento futuro. El tamaño adecuado de las particiones depende en gran medida de la cantidad de datos disponibles y la seguridad que se necesite en la estimación del error, por lo general, 80 % y 20 % para los datos de entrenamiento y de prueba, respectivamente, suele dar buenos resultados.

6. Por último, subdividimos los datos de entrenamiento y de prueba para tener por separados las características y la variable objetivo en ambos conjuntos.

El desarrollo de esta sección lo conseguimos en el Notebook de Jupyter *preparación_ datos* [3].

3.2. Creación de modelos

El paso siguiente a la preparación de los datos es seleccionar el algoritmo que se va a emplear, siendo ésta quizás la parte más difícil. Con el conjunto de datos preparados en la sección anterior se utilizan algunos de los principales algoritmos usados en Machine Learning, sin tomar en cuenta si es el más apropiado o no, el objetivo es compararlos e identificar el que mejor resultado obtiene al predecir si el paciente sufrirá o no un ataque cerebrovascular. Para esto se utilizó la librería Scikit_learn de Python [4] enfocada en el aprendizaje automático de código abierto que admite el aprendizaje supervisado y no supervisado.

El enfoque está basado completamente en el aprendizaje supervisado, los detalles y todo el código utilizado se encuentra en el Notebook de Jupyter *modelos* [8] donde se trabajaron los siguiente algoritmos.

1. Árbol de decisión (DecisionTree)

   Los árboles de decisión son modelos predictivos formados por reglas binarias, con las que se consigue repartir las observaciones en función de sus atributos y predecir así el valor de la variable respuesta. [18].

   Se utilizan sobre todo para clasificación de información, bifurcando y modelando los posibles caminos tomados y su probabilidad de ocurrencia para mejorar su precisión. El primer nodo se llama raíz, luego se descompone en ramas que se subdividen hasta llegar a las hojas que son los nodos finales. En particular, buscará la mejor ruta, balanceando la posibilidad de ocurrencia y su importancia en cada rama y hojas para clasificar un resultado.

   Pueden realizar tareas de clasificación o regresión[13].

   - Clasificación: Se aplica cuando la variable objetivo es categórica.
   - Regresión: Se aplica cuando la variable objetivo es continua.

   Los árboles de decisión pueden ser inestables porque pequeñas variaciones en los datos pueden resultar en la generación de un árbol completamente diferente. Además, te dejan con una decisión difícil, un árbol profundo con muchas hojas tiene mucha varianza, pero se ajustan mucho a los datos de entrenamiento, y es aquí donde vemos el sobreajuste porque cada predicción proviene de datos históricos de solo algunos datos en su hoja. Ahora bien, un árbol poco profundo con pocas hojas, tiene poca varianza, pero no consiguen representar la relaciones entre las variables, por lo tanto, tendrá un desempeño deficiente porque no logra capturar tantas distinciones en los datos brutos.

   Este método sufre el problema de equilibrio entre sesgo y varianza, esto se soluciona con los métodos de ensamble, los cuales buscan equilibrar.

2. Bosques aleatorios (Random Forest)

   Un modelo Random Forest es un tipo de ensamble en Machine Learning que combina un conjunto de árboles de decisión individuales, donde la predicción de una nueva observación se obtiene agregando las predicciones de todos los árboles individuales que forman el modelo.

   Al igual que los árboles de decisión, los bosques tienen dos subtipos: clasificación y regresión[14].

3. Vecinos cercano (Nearest Neighbors)

   La idea general de los métodos de Nearest Neighbors es encontrar un número predefinido de muestras de entrenamiento más cercanas a la que se está tratando de predecir y clasifica el punto de interés basado en la mayoría de datos que le rodean.

   El aprendizaje basado en vecinos supervisados se clasifica en: clasificación y regresión, donde éstos a su vez se clasifican en:

   - KNeighbors, el número de muestras puede ser una constante definida por el usuario.
   - RadiusNeighbors, el número de muestra varia según la densidad local de puntos.

   KNeighbors es la técnica más utilizada. La elección óptima del valor depende en gran medida de los datos, en general, un valor mayor suprime los efectos del ruido, pero hace que los límites de clasificación sean menos distintos. En los casos en los que los datos no se muestrean de manera uniforme, RadiusNeighbors puede ser una mejor opción[15].

4. Regresión lineal

   Estadísticamente, regresión lineal es una aproximación para modelar la relación entre una variable dependiente y y una o mas variables X.

   La regresión lineal es un algoritmo de aprendizaje supervisado, la idea es obtener automáticamente una "recta" que se busca con la tendencia de predicción de un conjunto de datos continuos. Para hacerlo, se mide el error con respecto a los puntos de entrada y el valor de salida real. El algoritmo deberá minimizar el coste de una función de error cuadrático y esos coeficientes corresponderán con la recta óptima[16].

   Para este modelo en particular, se utilizó el conjunto de datos 2 [6], el cual tiene información sobre los precios de las casas de la ciudad de Windsor (Canadá). Estos datos también fueron tratados antes de aplicar el modelo y tienen información sobre las variables:

   a) Las variables predictoras.
      - lotsize: Tamaño del lote
      - bedrooms: Número de cuartos
      - bathrooms: Número de baños
      - stories: Número de pisos, incluyendo el sótano
      - driveway: Si tiene pasador o no
      - recreation: Si tiene cuarto de recreación o no

- fullbase: Si tiene sótano o no
- gasheat: Si tiene calentador a gas o no
- aircon: Si tiene aire acondicionado o no
- garage: Si tiene garage o no

b) Variable objetivo

- price: Precio

3.3. Validación y optimización

Luego de desarrollar modelos de aprendizajes supervisados, se muestra como aptimizarlos y evaluarlos con la metodología adecuada.

El conjunto de datos trabajado en esta sección es uno de los disponibles en la librería Scikit-learn, el cual consta de 3 tipos diferentes de lirios (Setosa, Versicolour y Virginica) de longitud de pétalos y sépalos. Las columnas son: longitud del sépalo, ancho del sépalo, longitud del pétalo y ancho del pétalo [7].

1. Validación cruzada (Cross-validation)

   Consiste en entrenar y luego validar el modelo en varios cortes del conjunto de entrenamiento, el cual se divide en k conjuntos más pequeños y se les aplica el siguiente procedimiento:

   - Un modelo se entrena usando $k-1$ cortes como datos de entrenamiento.
   - El otro corte se utiliza como un conjunto de pruebas para calcular una medida de rendimiento como la precisión.

   Este enfoque puede ser computacionalmente costoso, pero no desperdicia demasiados datos, siendo una gran ventaja para muestras muy pequeñas[9].

2. Curva de validación (Validation Curve)

   Permite determinar los puntajes de entrenamiento y prueba para diferentes valores de parámetros[10].

3. Búsqueda de cuadrículas (GridSearchCV)

   Los distintos modelos de Machine Learning tienen uno o varios parámetros. Si queremos probar las mejores combinaciones en los parámetros para conseguir un mejor rendimiento, utilizamos la función búsqueda de cuadrículas, la cual hace una búsqueda exhaustiva de valores de parámetros especificados para un estimador[11].

4. Curva de aprendizaje (Learning Curve)

   Con las curvas de aprendizaje se podría saber si el modelo tiene un mejor rendimiento si se le proporciona más datos. Éstas muestran la evolución del desempeño en función de la cantidad de datos. Normalmente, mientras más datos, mejor es su rendimiento[12].

Los detalles y todo el código utilizado en esta sección se encuentra en el Notebook de Jupyter *optimizacion_ validacion_ evaluacion* [17].

	id	gender	age	hypertension	heart_disease	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Rural	174.12	24.0	never smoked	1
...	...	...	...	...	...	...	...	...	...	...
5105	18234	Female	80.0	1	0	Urban	83.75	NaN	never smoked	0
5106	44873	Female	81.0	0	0	Urban	125.20	40.0	never smoked	0
5107	19723	Female	35.0	0	0	Rural	82.99	30.6	never smoked	0
5108	37544	Male	51.0	0	0	Rural	166.29	25.6	formerly smoked	0
5109	44679	Female	44.0	0	0	Urban	85.28	26.2	Unknown	0

5110 rows × 10 columns

Figura 1: Datos originales del conjunto de datos 1

4. Resultados y Análisis

El conjunto de datos al momento de importarse, se visualiza como se muestra en la Figura (1). Luego de hacerle toda la limpieza y preparación, queda como se muestra en la Figura (2).

Finalmente, el conjunto quedó dividido como se muestra en el Cuadro 1, permitiendo modelar y ajustar los distintos algoritmos de Machine Learning supervizado. Los diferentes puntajes obtenidos en estos algoritmos se muestran en el Cuadro 2.

Cuadro 1: División del conjunto de datos 1

Categoría	Subcategoría
Entrenamiento	Características Variable objetivo
Prueba	Características Variable objetivo

En el Cuadro 2 se evidencia lo siguiente:

- Los mejores algoritmos para el conjunto de datos 1 fueron Árbol de clasificación y Random Forest, ambos con un puntaje de 0.955193482688391.
- El algoritmo de árbol de regresión tuvo una puntuación de -1.331750339213027, siendo no apropiado para el conjunto de datos 1, ya que los valores de la variable objetivo son discretos.
- El algoritmo de regresión lineal para el conjunto de datos 2, tuvo una puntuación de 0.6151361844552091, el cual no es un buen modelo, ya que para afirmar que lo sea, la puntuación debería estar entre 91 y 99 %, aproximadamente.

	gender	age	hypertension	heart_disease	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	0	67.0	0	1	0	228.69	36.6	0	1
2	0	80.0	0	1	1	105.92	32.5	1	1
3	1	49.0	0	0	0	171.23	34.4	2	1
4	1	79.0	1	0	1	174.12	24.0	1	1
5	0	81.0	0	0	0	186.21	29.0	0	1
...	...	...	...	...	...	...	...	...	...
5104	1	13.0	0	0	1	103.08	18.6	3	0
5106	1	81.0	0	0	0	125.20	40.0	1	0
5107	1	35.0	0	0	1	82.99	30.6	1	0
5108	0	51.0	0	0	1	166.29	25.6	0	0
5109	1	44.0	0	0	0	85.28	26.2	3	0

4909 rows × 9 columns

Figura 2: Datos tratados del conjunto de datos 1

Cuadro 2: Puntajes de los distintos algoritmos usados

Conjunto de datos	Algoritmo	Puntaje
1	Árbol de regresión Clasificación	0.955193482688391
	Árbol de regresión Regresión	-1.331750339213027
	Random Forest	0.955193482688391
	Nearest Neighbors	0.9501018329938901
2	Regresión lineal	0.6151361844552091
3	Nearest Neighbors	0.9

- El algoritmo para el conjunto de datos 3 fue el de Nearest Neighbors, el cual tuvo una puntuación de 0.9. Sin embargo, éste se mejora en la siguiente sección con un ajuste de parámetros.

En la sección 3 obtuvimos los siguientes resultados:

- Al hacer la validación cruzada en el conjunto de datos 3, el puntaje del algoritmo de Nearest Neighbors que en un principio era 0.9, sube a 0.96.
- En la curva de validación que se muestra en la Figura (3) se evidencia que el valor más óptimo para el parámetro de 'n_neighbors' es 10.
- La cuadrícula de búsqueda nos arroja la mejor combinación de parámetros para este algoritmo, la cual es 'metric': 'euclidean' y 'n_neighbors': 5, con una puntuación de 0.9833333333333334.
- En la curva de aprendizaje se observa que la cantidad de datos mínimas para un buen modelo es 20 como se observa en la Figura (4).

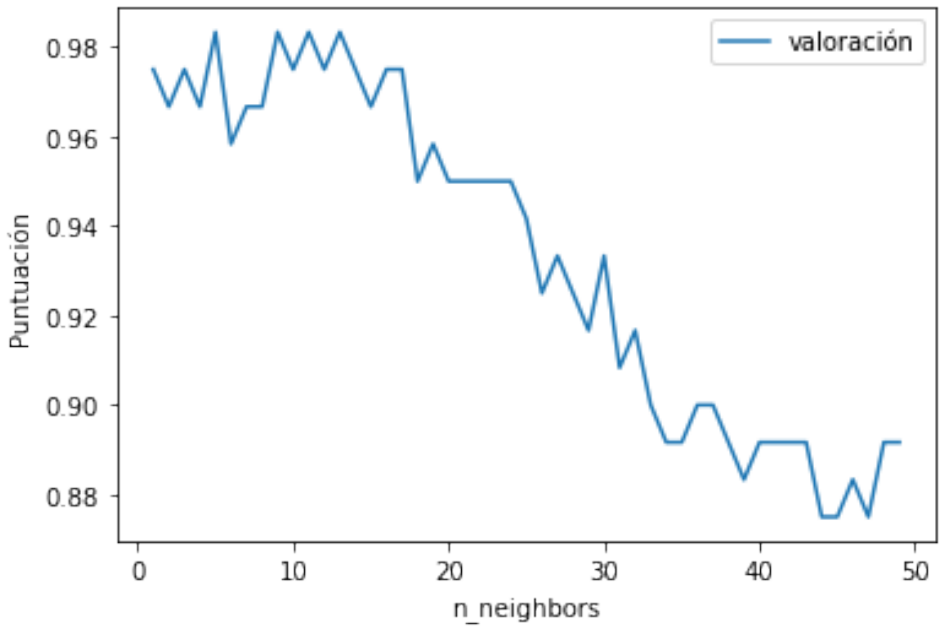

Figura 3: Curva de validación para el parámetro 'n_neighbors' del modelo Nearest Neighbors para el conjunto de datos 3

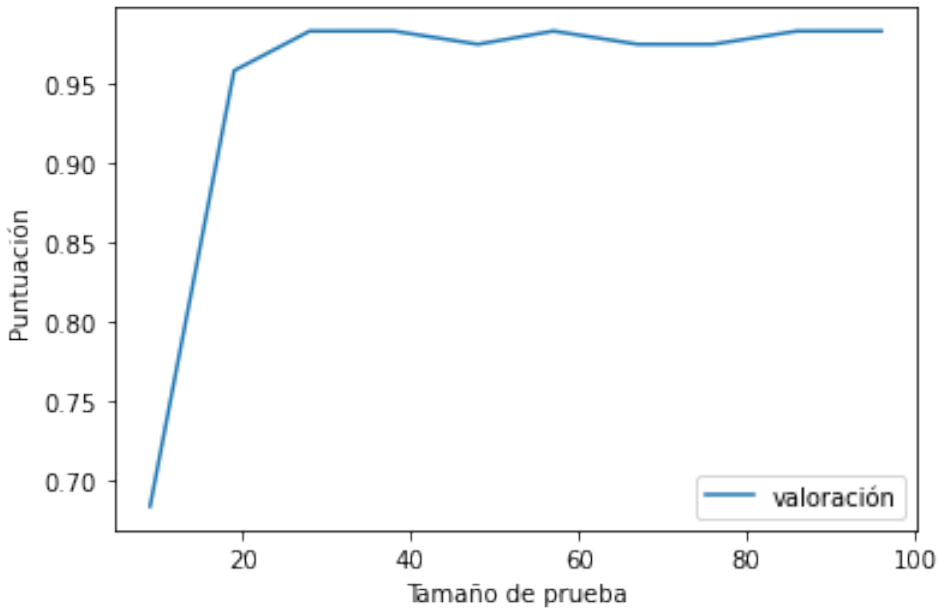

Figura 4: Puntuación del modelo Nearest Neighbors para el conjunto de datos 3 dependiendo del tamaño de la muestra

5. Conclusiones y Recomendaciones

Actualmente, por medio de la tecnología podemos intercambiar y acceder a millones de datos, y la capacidad de poder procesarlos es una de las cualidades más potentes del Machine Learning, ya que termina siendo de gran ayuda al desarrollo del ser humano, sus aplicaciones las encontramos en el ámbito económico, tecnológico y social de nuestras vidas. Sin embargo, la información que nos suministra aún depende en gran parte de la supervisión humana, debemos escoger el algoritmo que más se adapte al conjunto de datos, al igual que validar y ajustar los modelos dependiendo del objetivo final, debido a que todavía no son capaces de pensar por sí solas.

En este trabajo sólo se trabajó el aprendizaje supervisado, sin embargo, hay mucho por explorar, en especial las redes neuronales, donde su idea pricipal es imitar el funcionamiento de las redes neuronales de los organismos vivos, es decir, aprender por niveles conectados entre sí, y mientras más niveles haya, el algoritmo se vuelve mucho más complejo y es lo que conocemos como deep learning.

Referencias

[1] IBM. What is machine learning?? https://www.ibm.com/cloud/learn/machine-learning

[2] Pandas. https://pandas.pydata.org/

[3] Eliana Bonalde. Jun. 13, 2021. ProyectoModelado_ElianaBonalde. https://github.com/bonaldee/ProyectoModelado_ElianaBonalde/blob/main/codigo/preparacion_datos.ipynb

[4] Scikit-learn: Machine Learning in Python. https://scikit-learn.org/stable/

[5] Kaggle. Stroke Prediction Dataset. https://www.kaggle.com/fedesoriano/stroke-prediction-dataset?select=healthcare-dataset-stroke-data.csv

[6] Rdatasets. Available datasets. https://vincentarelbundock.github.io/Rdatasets/articles/data.html

[7] Scikit Learn. The Iris Dataset. https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html?highlight=data%20iris

[8] Eliana Bonalde. Jun. 13, 2021. ProyectoModelado_ElianaBonalde. https://github.com/bonaldee/ProyectoModelado_ElianaBonalde/blob/main/codigo/modelos.ipynb

[9] Scikit Learn. Cross-validation: evaluating estimator performance https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation

[10] Scikit Learn. sklearn.model_selection.validation_curve https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.validation_curve.html?highlight=validation_curve#sklearn.model_selection.validation_curve

[11] Scikit Learn. sklearn.model_selection.GridSearchCV https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html?highlight=gridsearchcv#sklearn.model_selection.GridSearchCV

[12] Scikit Learn. sklearn.model_selection.learning_curve. https://github.com/HarveyMaddocks/gofundme_webscraper

[13] Scikit Learn. Decision Trees. https://scikit-learn.org/stable/modules/tree.html#tree

[14] Scikit Learn. Forests of randomized trees. https://scikit-learn.org/stable/modules/ensemble.html#forest

[15] Scikit Learn. Nearest Neighbors. https://scikit-learn.org/stable/modules/neighbors.html#

[16] Scikit Learn. Linear Models. https://scikit-learn.org/stable/modules/linear_model.html

[17] Eliana Bonalde. Jun. 13, 2021. ProyectoModelado_ElianaBonalde. https://github.com/bonaldee/ProyectoModelado_ElianaBonalde/blob/main/codigo/optimizacion_validacion_evaluacion.ipynb

[18] Amat Rodrigo, Joaquín. Machine learning con Python y Scikit-learn. Aug, 2020. https://www.cienciadedatos.net/documentos/py06_machine_learning_python_scikitlearn.html