Modelo de predicción de masa de pingüinos

Autora: Andrea Medina Rico **Matrícula:** A01705541

Abstracto. Este reporte detalla la creación e implementación de un modelo para predecir la masa en kilogramos de un pingüino, de acuerdo a algunas de sus características. Para esto, el modelo fue entrenado y evaluado con una regresión lineal múltiple, de la mano con un algoritmo para su optimización. Este modelo reporta un coeficiente de determinación de 0.814, indicando la capacidad de explicar el 81.4% en la variación de la masa de los pingüinos.

1. Introducción

En el ámbito de la zoología, un modelo de predicción de la masa de un animal a partir de algunas de sus medidas corporales puede resultar una gran herramienta. Esto reduciría la necesidad de procesos invasivos y costosos para mantener registro de sus medidas y su salud.

El objetivo de este modelo es generar una predicción de la masa de un pingüino a partir de algunas de sus medidas y sus características. El conjunto de datos 'penguins_size.csv', presenta información de pingüinos de tres distintas especies dentro de la misma familia. Se busca demostrar que se puede realizar la predicción de una característica corporal (en este caso, la masa), sin importar que un animal pueda tener variaciones dependiendo de la raza o especie que sea.

2. Extracción, transformación y limpieza (ETL) de los datos

a. Descripción del conjunto de datos

La información de los pingüinos proviene del archivo 'penguins_size.csv'. Contiene información sobre pingüinos de tres islas distintas: Dream, Biscoe y Torgersen, las tres situadas en la Antártida. Además, distingue entre las tres especies de la familia Pygoscelis, conocida como "pingüinos de cola de cepillo". Estas son Adelia (Adelie), Gentú o Papúa (Gentoo) y Barbijo (Chinstrap).

Variables numéricas:

- *culmen_depth_mm:* Alto de la cresta superior del pico (culmen) medida en milímetros.
- *culmen_length_mm:* Longitud de la cresta superior del pico (culmen) medida en milímetros.
- *flipper_length_mm:* Longitud de la aleta medida en milímetros.
- *body_mass_g:* Masa del pingüino medida en gramos.

Variables categóricas:

- *sex:* Sexo (macho o hembra).
- *species:* Especie del pingüino.

- *island:* Isla de proveniencia.

b. Transformación de los datos

Primeramente, se eliminaron los ejemplos con datos faltantes o nulos, debido a que eran 11, una cantidad baja.

Las variables categóricas, al ser de clasificación en lugar de numéricas, no pueden ser directamente procesadas por el modelo. Por ello, se procesaron de dos maneras distintas.

La columna hembra podía tener dos valores distintos, 'male' o 'female'. Para transformarla a numérica, se le dio un valor binario donde todos los machos corresponden a 1 y las hembras a 0.

En el caso de la especie y la isla, ambas columnas pueden tener tres valores distintos. Una transformación numérica como en el caso anterior no es lo más ideal, debido a que el modelo comienza a asociarlo con una relación lineal, donde una categoría es "mayor o menor" que otra. Por ello, se aplicó one-hot encoding. Se crearon k - 1 nuevas columnas binarias, siendo k el número de valores únicos en la columna a dividir. Estas nuevas columnas se conocen como "variables dummy".

Al generar k - 1 columnas, se evita caer en la trampa donde dos de las variables dummy tienen una colinealidad perfecta. Se toma como base la categoría no incluida (si todas las columnas son 0, corresponde a la categoría base). Se añadieron *species Adelie, species Gentoo, island Dream* e *island Biscoe*.

En las transformaciones numéricas, se realizó una conversión de unidades para que la diferencia entre las categorías no fuera tan grande. Los gramos pasaron a kilogramos y los milímetros a centímetros. Con esto, las columnas también fueron renombradas acorde con el cambio.

c. Detección de anomalías

Para observar el comportamiento de cada variable numérica, se realizó un histograma por cada una. Esto demostró que no existía ningún valor atípico que sesgara los datos, pero sí demostraron la tendencia de los datos a repartirse en dos grupos.

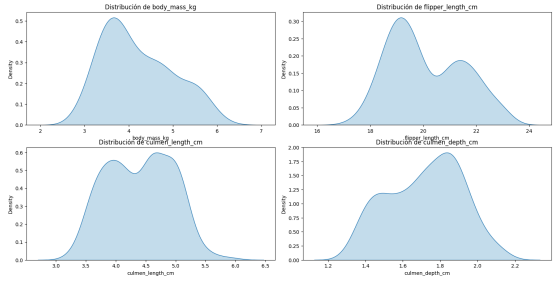

Figura 1. Distribución de variables numéricas

La distribución de los datos se aglomera tomando como media dos puntos distintos. Esto podría deberse a un solapamiento entre las mismas especies al ser estas parte de la misma familia. Por ejemplo, en una categoría, dos especies podrían presentar una característica similar, mientras que la tercera tiene valores distintos.

3. Selección de variables para el modelo

a. Correlación de las variables

Para mayor entendimiento sobre cómo se relaciona la masa de los pingüinos con el resto de las categorías, se realizó una matriz de correlación. En cada celda, muestra un número entre -1 y 1, donde -1 es una fuerte relación negativa y 1, una fuerte relación positiva.

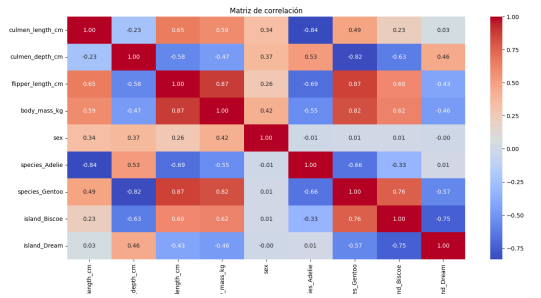

Figura 2. Matriz de correlación entre variables

En la fila o columna de *body_mass_kg*, se observa que, en general, tiene relaciones fuertes y medianas con la mayoría de las variables. Al no haber ninguna con correlación baja ni que se considere no importante para la predicción, se decidió conservarlas todas.

b. Dispersión de las variables

La relación entre la variable a predecir (la masa) contra cada una de las categorías se visualiza en la siguiente figura:

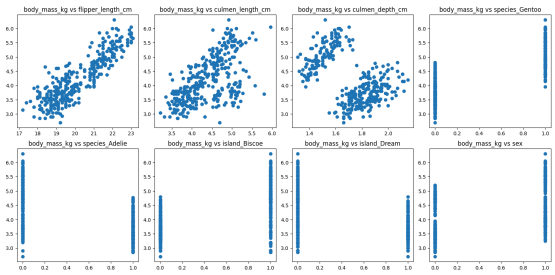

Figura 3. Diagramas de dispersión de las categorías selectas contra la masa en kg.

Se observa una relación cercana a lineal en las tres variables numéricas. El comportamiento de las demás gráficas se debe a que son los valores binarios (0 y 1) provenientes de las variables categóricas transformadas (*sex*, *species* e *island*). A pesar de no ser tan claro como en las numéricas, se observa una clara distinción de los valores de la masa cuando pertenecen o no a una categoría, es decir, cuando su valor es 0 o 1.

4. División de los datos

a. Train y test

Para que el modelo tenga el comportamiento esperado, debe someterse a entrenamiento. Para asegurar que el modelo entrenado es funcional, debe someterse a pruebas. Realizar ambas fases con el mismo conjunto de datos es erróneo debido a que se entrena y prueba con la misma información que el modelo ya conoce. Genera una idea falsa de que el modelo es acertado en todos los casos. Al probarlo con nuevos datos, lo más probable es que caiga en un problema donde no se acerca a los valores a menos que sean aquellos con los que entrenó, conocido como overfitting.

Por esta razón, se realizó una división aleatoria de los datos en porcentaje 80% para entrenamiento y 20% para pruebas. El conjunto de datos, tras la limpieza, consta de 333 ejemplos. Por lo que a train lo constituyen 266 registros y a test, 67.

b. Validation

Antes de que el modelo entrenado pase a la fase final de pruebas, se tiene un apartado de datos para validar los resultados y regresar a ajustar los hiper parámetros en el entrenamiento en caso de ser necesario. En esta situación, la cantidad de instancias es relativamente baja, por lo que destinar un porcentaje de ejemplos a esta etapa podría resultar perjudicial para el modelo. Por lo mismo, la validación se realizará con los datos de entrenamiento mediante una técnica de validación cruzada, la cual se abordará más adelante.

5. Construcción del modelo

El objetivo del modelo es predecir el valor de la masa de los pingüinos con base en un grupo de categorías. Para lograrlo, debe someterse a un entrenamiento y ajuste hasta que los resultados se acerquen a lo esperado, con ayuda de distintas funciones y algoritmos.

a. Función de hipótesis

Este modelo sigue una regresión lineal múltiple para explicar la relación entre la variable a predecir (la masa del pingüino) respecto a las variables independientes seleccionadas.

$$y = \beta_0 + \sum_{i=1} \beta_i x_i \cdot$$

Figura 4. Función de hipótesis para regresión lineal múltiple.

La función de hipótesis es la ecuación lineal que describe el peso de cada variable para llegar al resultado final. El resultado de esta ecuación es la predicción de la variable dependiente.

c. Función de costo

La función de costo, también conocida como función de pérdida o función de error, compara el valor predicho por la función de hipótesis contra el valor real de nuestra variable a predecir (en este caso, la masa del pingüino). Como su nombre lo dice, mide el error o la pérdida que hubo. Este valor es de suma importancia, debido a que nos ayuda a evaluar el desempeño del modelo y saber qué tan cerca estamos matemáticamente de realizar una predicción acertada. El objetivo de este modelo será acercarlo a 0, indicando que la mayoría de los valores son acertados o tienen poco rango de error. Se utilizan las dos ecuaciones siguientes:

$$\mathrm{MSE} = \frac{\Sigma(y_i - p_i)^2}{n}$$

Figura 5. Función del Mean Squared Error.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Figura 6. Función del Mean Absolute Error.

La primera se utiliza en el algoritmo de optimización, para calcular el cambio en los parámetros necesario para minimizar el error. La segunda, se utiliza como error promedio interpretable en las unidades de medida de nuestra variable a predecir, en este caso, en kilogramos.

d. Gradiente descendiente

La manera en la que el modelo minimizará su error es utilizando el algoritmo de optimización gradiente descendiente (gradient descent) para su entrenamiento. Este algoritmo encuentra los parámetros de la función de hipótesis necesarios para minimizar lo más posible la función de costo. Se define como:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

Figura 7. Función para calcular el nuevo valor de un parámetro con gradiente descendiente.

Para calcular el valor de un nuevo parámetro de la función de hipótesis (la ecuación lineal múltiple), a su valor actual se le resta el gradiente de la función de costo. Desarrollada para una regresión lineal múltiple, queda de la siguiente forma:

$$\Theta_j := \Theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}[(h_\Theta(x_i) - y)x_i]$$

Figura 8. Función del gradiente descendiente desarrollada para regresión lineal.

e. Validación cruzada

El algoritmo de validación cruzada (cross validation) se utiliza para probar la eficiencia del modelo, con el objetivo de ajustar los hiper parámetros y mejorar el modelo. Se ajusta con este modelo debido a las pocas instancias en el conjunto de datos.

Del 80% de datos destinados a entrenamiento, se divide en k subpartes (en este caso 10). En cada iteración, se toman 9 partes para entrenar y una para validar, repitiendo hasta usar todos los datos para validación. Esto permite evaluar el error arrojado y ajustar los hiper parámetros, como el número de épocas a repetir el algoritmo de optimización o el learning rate (el tamaño de los pasos al optimizar).

f. Normalización de los datos

La normalización de los datos es una técnica de escalamiento, la cual estandariza el rango de valores de cada característica. Esto evita que el modelo se sesgue a otorgarle mayor peso a las variables de valores más altos. Permite que cada variable tenga una contribución proporcional a la ecuación.

En este modelo, utilizaremos la estandarización o z-score, debido a que es comúnmente utilizado en regresiones lineales y no resulta tan afectado por valores extremos. Resulta favorable considerando la distribución de las categorías, aglomerada en dos grupos y con un rango de valores considerable. Está dada por la ecuación:

$$Z = \frac{X - \mu}{\sigma}$$

Figura 9. Función de estandarización de datos.

La estandarización se realiza tomando la media y desviación estándar por categoría del conjunto de datos destinados para entrenamiento. Se aplica a todas las categorías tanto de train como de test, a excepción de los valores predichos y por predecir.

g. Evaluación del modelo

Los resultados del modelo se evaluarán con el coeficiente de determinación R^2, un valor entre 0 y 1 que indica cuántos datos es capaz de explicar el modelo. Al multiplicarse por 100, es interpretable para nosotros como porcentaje. Se calcula como:

$$R^2 = \frac{\sum_{t=1}^{T} (\hat{Y}_t - \overline{Y})^2}{\sum_{t=1}^{T} (Y_t - \overline{Y})^2}$$

Figura 10. Función del coeficiente de determinación.

6. Entrenamiento del modelo

Cada función y algoritmo revisado anteriormente se aplica para el entrenamiento de este modelo.

a. Entrenamiento y validación

Primeramente, se tomó el conjunto de datos de entrenamiento para introducirlo en el algoritmo de validación cruzada. La división de grupos en este algoritmo fue de k = 10 clases. Dentro de este algoritmo, se indicó que por cada una de las 10 pasadas, se tendría un número de épocas y learning rate específico.

En cada iteración de cv, los datos fueron estandarizados con la media y desviación estándar calculadas de las 9 subpartes de entrenamiento.

Tras un conjunto de pruebas y análisis del comportamiento del error con MAE, se decidió que los hiper parámetros finales serían un learning rate de 0.001 y un número de épocas de 5000.

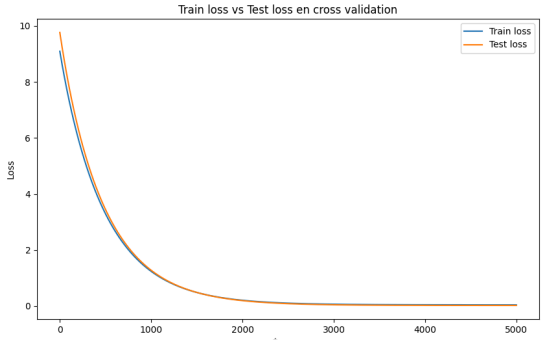

Figura 11. Pérdida de entrenamiento contra la de validación en cross validation.

La gráfica anterior indica que la pérdida encontrada en el entrenamiento y en la validación es bastante similar, por lo que no se detecta algún problema de sobre ajuste o falta del mismo. Además, se observa que, a partir de la época 3000, no existió una reducción tan drástica del error y se mantuvo cercano a cero. Sin embargo, se decidió mantener hasta la época 5000 para reducir lo más posible el error. En este contexto, un MAE de 0.1 implica un error de 100 gramos. Habiendo masa de pingüinos de un rango entre 2.7 a 6.3 kilogramos, los decimales son bastante importantes.

El MAE tuvo un valor de 0.904 en entrenamiento y 0.906 en validación. Esto se interpreta como que en entrenamiento, se perdieron en promedio 904 gramos en los datos y, en la validación, hubo pérdida de 906 gramos en promedio. Es un valor relativamente alto. Sin embargo, se debe considerar que es el promedio de la pérdida de los modelos entrenados con una menor cantidad de datos, siendo 239 para el entrenamiento y 27 para la validación. Agregar más épocas no tendría un gran impacto a partir del 5000, por lo que se utiliza este número para el entrenamiento final.

Con esto en mente, se realizó el entrenamiento con los datos completos destinados a train (el 80%). Los datos de train y test fueron normalizados con la media y desviación estándar proveniente de train.

7. Evaluación del modelo

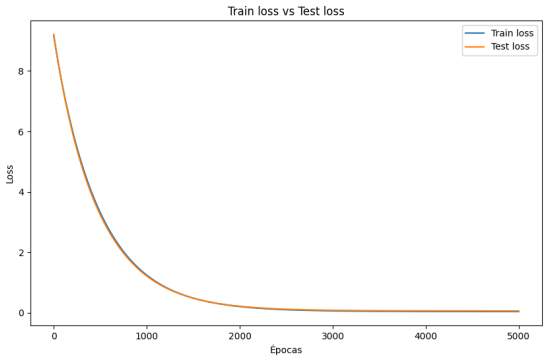

Figura 12. Pérdida de entrenamiento contra la pérdida de pruebas.

La gráfica anterior muestra un comportamiento de pérdida bastante similar entre el entrenamiento y las pruebas, tal que ambas líneas se solapan. El MAE dio un valor de 0.229 en el entrenamiento y 0.290 en las pruebas. Esto quiere decir que, en promedio, la pérdida de datos fue de 229 gramos en entrenamiento y 290 gramos en pruebas.

El coeficiente de determinación R^2 tiene un valor de 0.814 en los valores de pruebas. En las pruebas, el modelo fue

capaz de explicar el 81.4% del comportamiento de los datos.

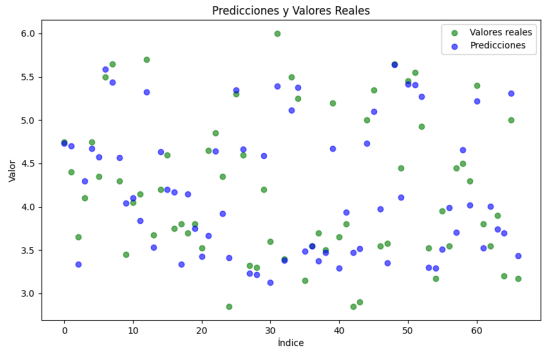

Figura 13. Comparación de valores predichos del modelo contra los valores reales.

En el diagrama de dispersión, se plasma el comportamiento de los valores predichos y reales sobre la masa de los pingüinos. A simple vista, se observa que tienen una varianza y comportamiento similar. Una gran parte de los puntos está sobrepuesto, indicando la predicción correcta, o muy cercano, indicando un error mínimo que no sobrepasa los 200 gramos. En el resto de puntos, aunque son menos, se observa un error mayor. Este caso se presenta más en los pesos reales muy bajos o muy altos, fuera del promedio. Esto puede indicar que al modelo le cuesta trabajo adaptarse a valores extremos.

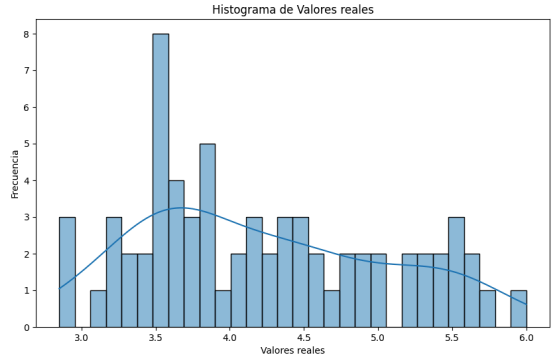

Figura 14. Histograma y distribución de los valores reales de masa (pruebas).

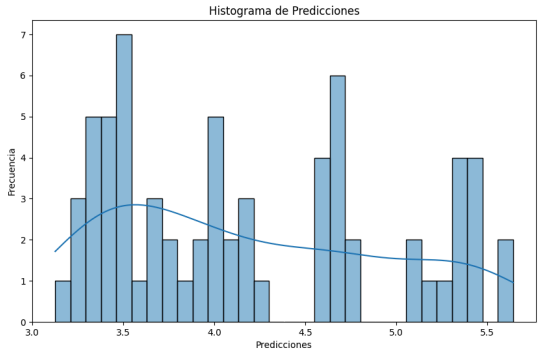

Figura 15. Histograma y distribución de los valores predichos de masa (pruebas).

Ambas gráficas anteriores demuestran dos puntos importantes. La cantidad de valores repetidos no es la misma, existe una varianza evidente. Esto se explica con el coeficiente de determinación, pues hay un 18.6% de datos que nuestro modelo no alcanza a predecir adecuadamente, por lo que el modelo no es completamente preciso. Sin embargo, se observa una distribución bastante similar en ambas. Esto quiere decir que el modelo sí aprende y sigue la tendencia general de los datos.

8. Referencias

Bobbitt, Z. (2021, 2 febrero). What is the Dummy Variable Trap? (Definition & Example). Statology. https://www.statology.org/dummy-variable-trap/

3.1. Cross-validation: evaluating estimator performance. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/cross_validation.html

Brownlee, J. (2016, 21 junio). How to choose the right test options when evaluating machine learning algorithms. Machine Learning Mastery. https://machinelearningmastery.com/how-to-choose-the-right-test-options-when-evaluating-machine-learning-algorithms/

Wagavkar, S. (2024, 20 noviembre). Introduction to the Correlation Matrix. Built In. https://builtin.com/data-science/correlation-matrix

Donges, N. (2024, 1 agosto). Gradient Descent in Machine Learning: A Basic Introduction. Built In. https://builtin.com/data-science/gradient-descent

Validación cruzada y k-fold cross-validation | Codificando Bits. (2024b, febrero 25). Codificando Bits. https://codificandobits.com/blog/validacion-cruzada-k-fold-cross-validation/

Jaiswal, S. (2024, 4 enero). What is normalization in machine learning? A comprehensive guide to data rescaling. Datacamp. https://www.datacamp.com/tutorial/normalization-in-machine-learning

Bergmann, D., & Stryker, C. (2025). Loss Function. IBM. https://www.ibm.com/think/topics/loss-function

Stewart, & Ken. (2025, 19 julio). Mean squared error (MSE) | Definition, Formula, Interpretation, & Facts. Encyclopedia Britannica. https://www.britannica.com/science/mean-squared-error

Siddiqui, L. (2025, 8 julio). Coeficiente de determinación: lo que nos dice el R cuadrado. Datacamp. https://www.datacamp.com/es/tutorial/coefficient-of-determination