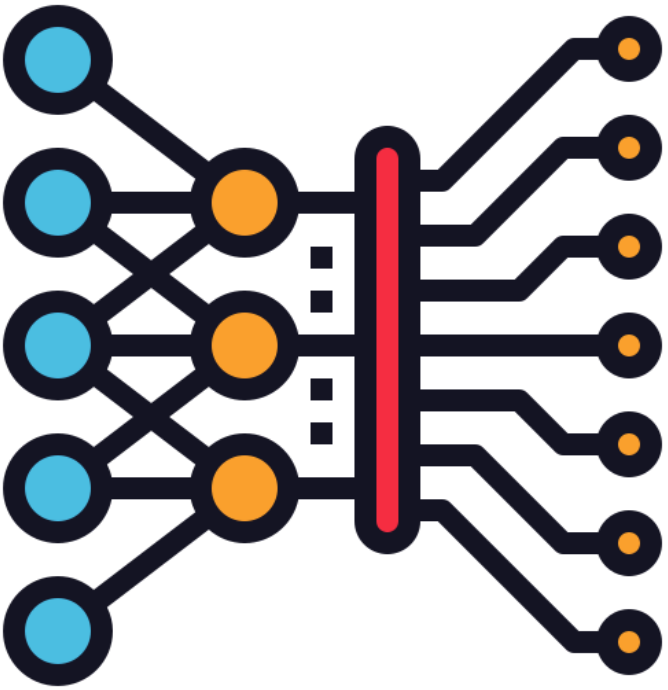
UNR Universidad Nacional de Rosario

Trabajo práctico Aprendizaje Automático

Tecnicatura Universitaria en Inteligencia Artificial



Docentes

- Joel Spak
- Agustín Almada
- Bruno Cocitto

FCEIA FACULTAD DE CIENCIAS EXACTAS INGENIERIA Y AGRIMENSURA

Alumnos

- Peroni Antonio
- Raffaeli Taiel

Año 2023

Consignas

Objetivo

Familiarizarse con la librería scikit-learn y las herramientas que brinda para el pre-procesamiento de datos, la implementación de modelos y la evaluación de métricas, y con TensorFlow para el entrenamiento de redes neuronales.

Actividades

1. Armar grupos de hasta dos personas para la realización del trabajo práctico. Dar aviso al cuerpo docente del equipo. En caso de no tener compañero, informar al cuerpo docente.

2. Realizar un análisis descriptivo, que ayude a la comprensión del problema, de cada una de las variables involucradas en el problema detallando características, comportamiento y rango de variación.

Debe incluir:

- Análisis y decisión sobre datos faltantes
- Visualización de datos (por ejemplo histogramas, scatterplots entre variables, diagramas de caja)
- ¿Está balanceado el dataset?
- Codificación de variables categóricas (si se van a utilizar para predicción).
- Matriz de correlación
- Selección de características para la predicción.
- Estandarización de datos.

3. Implementar la solución del problema de regresión con regresión lineal múltiple.
   - Probar con el método LinearRegression.
   - Probar con métodos de gradiente descendiente.
   - Probar con métodos de regularización (Lasso, Ridge, Elasticnet).

   - Obtener las métricas adecuadas (entre R2 Score, MSE, RMSE, MAE, MAPE).

4. Implementar la solución del problema de clasificación con regresión logística.
   - Obtener las métricas adecuadas (entre Accuracy, precision, recall, F1 Score, ROC-AUC, entre otras).

5. Implementar un modelo base para clasificación y uno para regresión.

6. Implementar las soluciones con una red neuronal.
    - Obtener las métricas adecuadas.

7. Optimizar la selección de hiperparámetros.
    - Probar validación cruzada.
    - Utilizar grid search, random search u optuna. Justificar su uso.

8. Implementar explicabilidad del modelo.
    - Utilizar SHAP o similar.

9. MLOps (a definir).
10. Escribir una conclusión del trabajo

Resolución de actividades

Dataset

Descripción y objetivo

El dataset se llama weatherAUS.csv y contiene información climática de Australia de los últimos diez años, incluyendo si para el día siguiente llovió o no y la cantidad de lluvia en las columnas 'RainTomorrow' y 'RainfallTomorrow'. El objetivo es la predicción de estas dos variables en función del resto de las características que se consideren adecuadas.

Tiene una columna 'Location' que indica la ciudad y el objetivo es predecir la condición de lluvia en las ciudades de Sydney, SydneyAirport, Canberra, Melbourne y MelbourneAirport (costa sureste).

Selección de características claves y datos incompletos

Luego de hacer un análisis simple de los datos que contiene el dataset se realizó un filtro de los mismos, dejando la mayor cantidad posible de datos para tratar de maximizar la obtención de información por parte de los modelos.

Primero se filtraron las localidades y luego se eliminaron las columnas de localidad (por no aportar mucho) y fecha ya que consideramos que se podría filtrar datos (data leakage).

Además el dataset presentaba datos incompletos:

Características // cantidad de datos faltantes

MinTemp	491	Humidity9am	570
MaxTemp	486	Humidity3pm	525
Rainfall	787	Pressure9am	735
Evaporation	1706	Pressure3pm	726
Sunshine	1948	Cloud9am	2677
WindGustDir	1457	Cloud3pm	2932
WindGustSpeed	1455	Temp9am	505
WindDir9am	695	Temp3pm	496
WindDir3pm	290	RainToday	787
WindSpeed9am	261	RainTomorrow	787
WindSpeed3pm	249	RainfallTomorrow	787

Los cuales se rellenaron bajo los siguientes criterios:

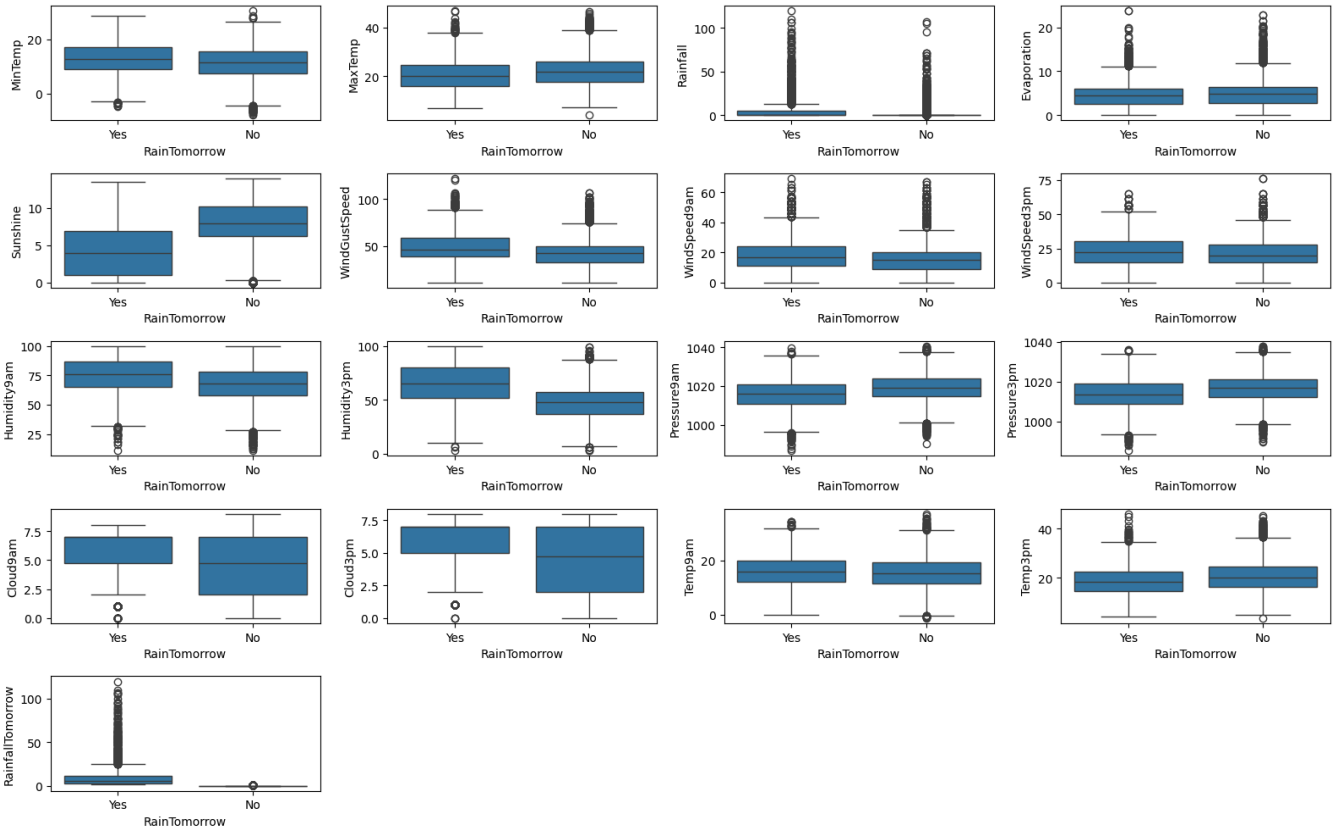
- Si la variable es numérica y continua, se rellena con el promedio.
- Si la variables es numérica y discreta, se rellena con la moda.
- Si la variables es categórica, se rellena con la moda.

Se eliminaron las filas incompletas de RainTomorrow y RainfallTomorrow ya que son variables que queremos predecir, por eso al tener datos faltantes no podemos inventarnos valores o repetirlos, debemos eliminar esas filas ya que no hay forma de rellenarlos.

Análisis descriptivo

Se buscó en un análisis descriptivo profundo una correlación entre variables y explicabilidad de los datos.

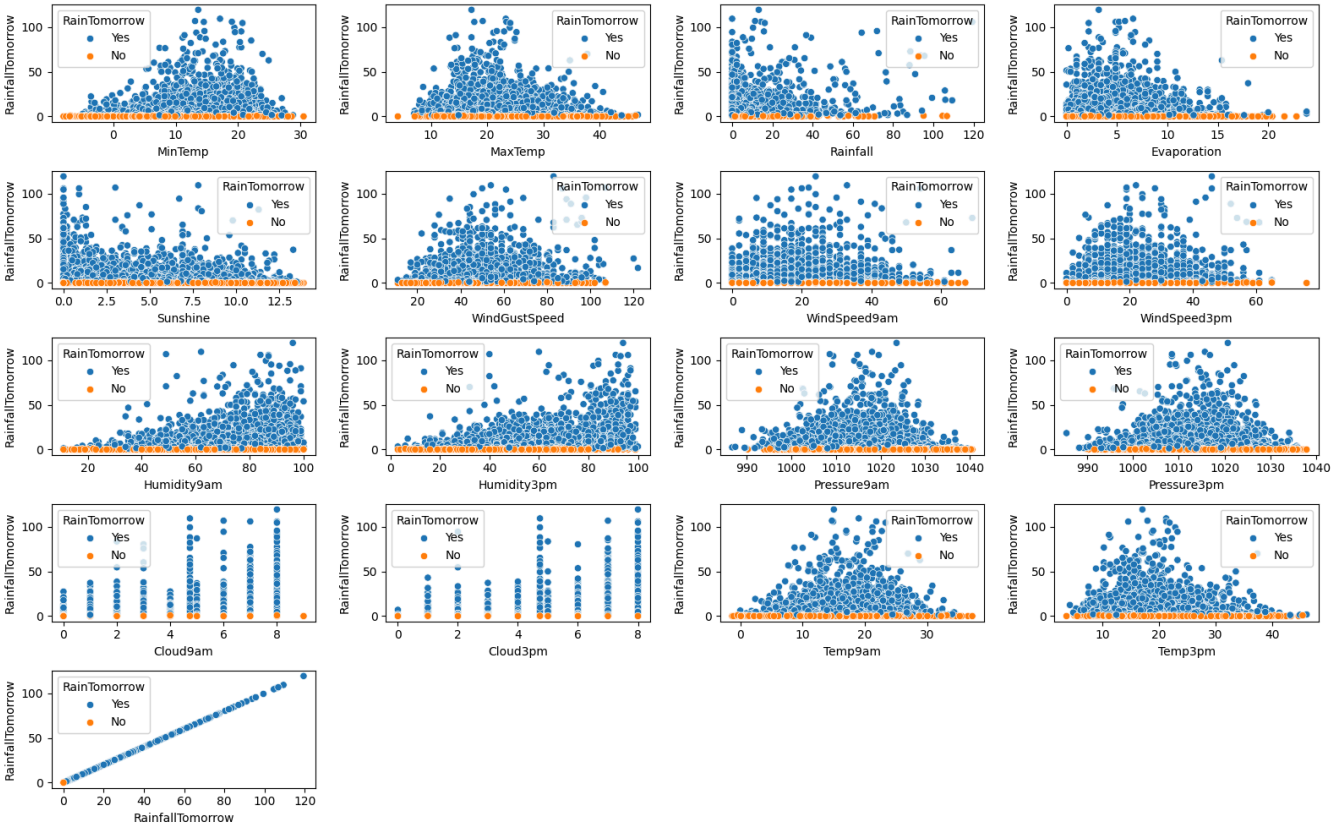
Gráfico de boxplot



En este gráfico podemos ver la explicabilidad entre las variables cuando llueve el día siguiente y cuando no con la variable a predecir “RainTomorrow” (categórica). Este tipo de análisis nos muestra si los datos se comportan o agrupan de una forma

diferente brindando más información de las variables y haciendo posible una diferenciación mayor en los casos de clasificación. Sin embargo se encuentran pocas variables que tienen buena explicabilidad en el conjunto de datos (como lo son las variables “Sunshine”, “Humidity9am”, “Humidity3pm”, “Cloud9am”, “Cloud3pm”).

Gráfico de dispersión

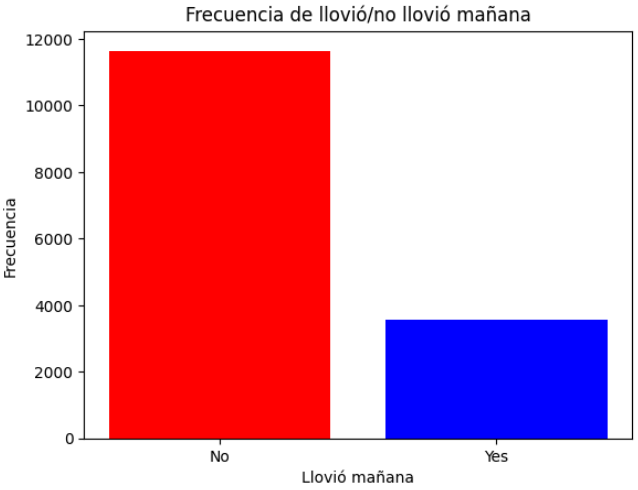


En el gráfico de dispersión se buscó ver una correlación lineal entre las variables con respecto a la variable a predecir “RainfallTomorrow” (numérica). Este tipo de gráfico da información clara de una tendencia para poder predecir un valor numérico en casos de regresión. Sin embargo algunas de las variables tienden a presentar una pequeña correlación lineal, como lo son “Sunshine”, “Humidity9am”, “Humidity3pm”, “Cloud9am”, “Cloud3pm”, “Rainfall”, “Evaporation ”, entre otras.

Se realizaron otros gráficos pero estos dos últimos (boxplot y dispersión) dieron información significativa acerca del comportamiento de las variables y cuales son las que podrían aportar más a la hora del entrenamiento de los modelos en los casos de clasificación como los de regresión.

También se analizó la frecuencia de los casos de lluvia ya que intentamos predecir cuándo lloverá.

Logramos ver un gran desbalance, esto se podría corregir con técnicas avanzadas de balanceo de datos a la hora de entrenar los modelos para que no afecte tanto su desempeño. Es importante tener un balance entre las variables a predecir para evitar que los modelos tengan un sobre ajuste.



División de datos

Para lograr una correcta división de datos y estandarización (utilizando StandardScaler) de los mismos decidimos crear dos conjuntos de datos para los casos de regresión y clasificación.

De esta forma se asegura la correcta división y estandarización de datos para cada entrenamiento y prueba de modelos según el problema a resolver.

Modelos de regresión lineal

Regresión lineal múltiple

La regresión lineal múltiple es una técnica estadística utilizada para analizar la relación entre una variable dependiente y dos o más variables independientes.

Su objetivo es encontrar los coeficientes que minimizan la diferencia entre los valores predichos por el modelo y los valores reales observados. Esto se logra ajustando el modelo utilizando métodos estadísticos.

Funcionamiento del modelo

1. Definición del Modelo Inicial:
    - Se inicializa un modelo con la ecuación general de regresión lineal múltiple.

2. Definición de la Función de Costo:

- La función de costo mide la discrepancia entre los valores predichos por el modelo y los valores reales observados. La función de costo comúnmente utilizada es la suma de los cuadrados de los residuos (diferencias entre los valores reales y predichos).

3. Minimización de la Función de Costo:
   - Se buscan los valores de los coeficientes que minimizan la función de costo.
   - Generalmente se utiliza el método de mínimos cuadrados.

4. Obtención de los Coeficientes Óptimos:
   - Los valores óptimos de los coeficientes se pueden obtener utilizando técnicas como el descenso de gradiente o soluciones analíticas según la complejidad del modelo y el conjunto de datos.

5. Validación del Modelo:
   - Una vez ajustado el modelo, se realiza la validación utilizando datos de prueba o técnicas de validación cruzada para asegurarse de que el modelo generalice bien a nuevos datos no vistos.

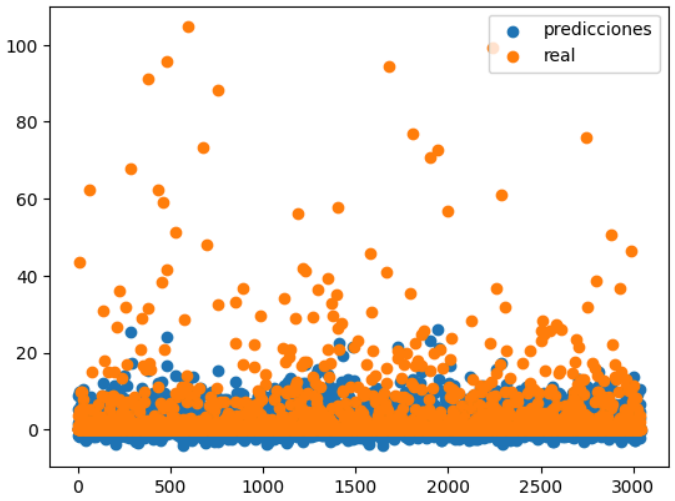
Aplicación del modelo

Luego de aplicar el modelo de regresion lineal multiple en el conjunto de datos se obtuvieron las siguiente métricas:

Error cuadrático medio en entrenamiento: 40.882791420013014
Error cuadrático medio en predicciones: 48.78116395095341

R2: 0.24110259259460842

En este caso la métrica de error MSE no varía tanto entre entrenamiento y prueba por lo tanto podría decirse que el modelo no tiene un sobreajuste.



En tanto a la métrica R2 el modelo solo puede explicar 24% de la variabilidad en los datos siendo esta muy baja.

En el gráfico de dispersión podemos ver las predicciones hechas por el modelo y los valores reales, los casos más extremos de lluvia no los capta el modelo.

Regresión lineal múltiple con gradiente estocástico

Esta regresión lineal múltiple es una variante de la original ya que utiliza una técnica de optimización llamada gradiente estocástico. El gradiente descendiente clásico de una función es un vector que apunta en la dirección de la mayor derivada de la función en un punto dado. Para minimizar la función, nos movemos en la dirección opuesta al gradiente, acercándonos al mínimo, que puede ser global o local, donde el gradiente es cero.

El gradiente descendente estocástico también busca optimizar en la dirección opuesta al gradiente, pero en lugar de usar todo el conjunto de datos para calcular el gradiente, se toma un dato a la vez.

Los datos se ordenan de manera aleatoria, y se calcula y actualiza el gradiente para cada muestra, avanzando a través de todas ellas antes de completar una época. Este enfoque, llamado estocástico, "muestrea" el dataset, siendo menos costoso en términos de cálculos de error y gradiente por muestra en lugar de todo el conjunto.

Aunque esto acelera la optimización al comenzar con la primera muestra, la desventaja es que el uso de un solo dato puede generar una exploración más ruidosa del problema.

Funcionamiento del modelo

1. Inicialización de Parámetros:
    - Se inicializan los coeficientes del modelo de regresión lineal múltiple.

2. Elección de Hiperparámetros:
    - Se eligen hiperparámetros importantes, como la tasa de aprendizaje (controla la magnitud de los ajustes realizados a los coeficientes) y el número máximo de iteraciones (independientemente de si ha convergido o no el modelo).

3. Iteraciones:
    - Por cada iteración:
        - Se selecciona aleatoriamente un ejemplo de entrenamiento del conjunto de datos.
        - Se calcula la predicción del modelo para ese ejemplo.
        - Se calcula el error entre la predicción y el valor real.

- Se actualizan los coeficientes utilizando la regla de actualización del gradiente estocástico.
- Se repiten estos pasos para cada ejemplo de entrenamiento.

4. Actualización de Coeficientes (Regla del Gradiente Estocástico):
   - La regla de actualización de los coeficientes para el gradiente estocástico es similar a la del descenso de gradiente clásico, pero se aplica a un solo ejemplo en lugar de todo el conjunto de datos. La actualización se realiza para cada coeficiente calculado para el ejemplo actual.

5. Convergencia:
   - El proceso de iteración se repite hasta que se alcanza un número máximo de iteraciones o hasta que se cumple un criterio de convergencia, como la convergencia del error.

6. Predicción:
   - Una vez que el modelo ha sido entrenado, se pueden realizar predicciones utilizando los coeficientes ajustados.

Aplicación del modelo

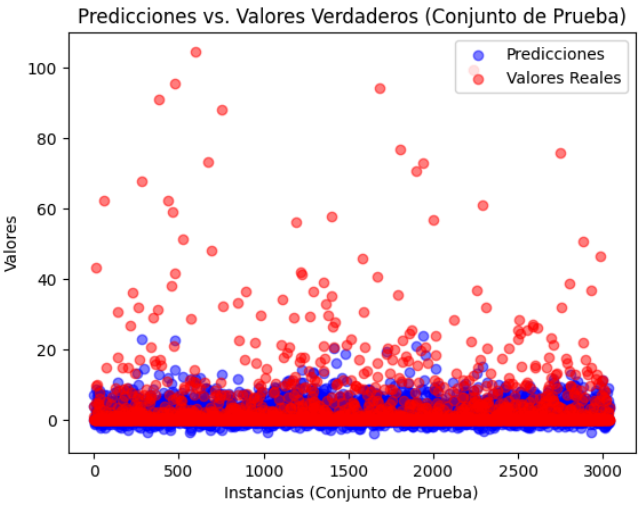
Error cuadrático medio en entrenamiento: 41.654097396635535
Error cuadrático medio en predicciones: 49.91914881040853

R2: 0.22339875591749692

El modelo se ajusta peor a los datos dando una menor explicabilidad de variabilidad según la métrica de R2.

Además la métrica MSE da un valor más elevado.

Esto podría mejorar si se eligieran mejores hiperparametros para el modelo.



Regresión lineal múltiple con gradiente batch

Este modelo es otra variante, pero en este caso no usamos todo el conjunto de datos ni solo una muestra a la vez; en su lugar, empleamos un batch (lote). El dataset se divide en n batches, y el gradiente se calcula para un batch a la vez, ajustando el vector. Este proceso se repite para cada batch en sucesión.

Después de una época, se vuelven a utilizar los distintos batches, asegurándonos de que contengan datos diferentes. Este enfoque converge más rápidamente que el descenso de gradiente clásico (GD) y presenta menos ruido que el descenso de gradiente estocástico (SGD).

El tamaño del batch es un hiperparámetro que debe ajustarse: si es muy grande, el método es similar a GD, y si es muy pequeño, es similar a SGD.

Funcionamiento del modelo

1. Inicialización de Parámetros:
    - Se inicializan los coeficientes del modelo de regresión lineal múltiple.

2. Elección de Hiperparámetros:
    - Se eligen hiperparámetros importantes, como la tasa de aprendizaje, el número máximo de iteraciones y el tamaño del batch.

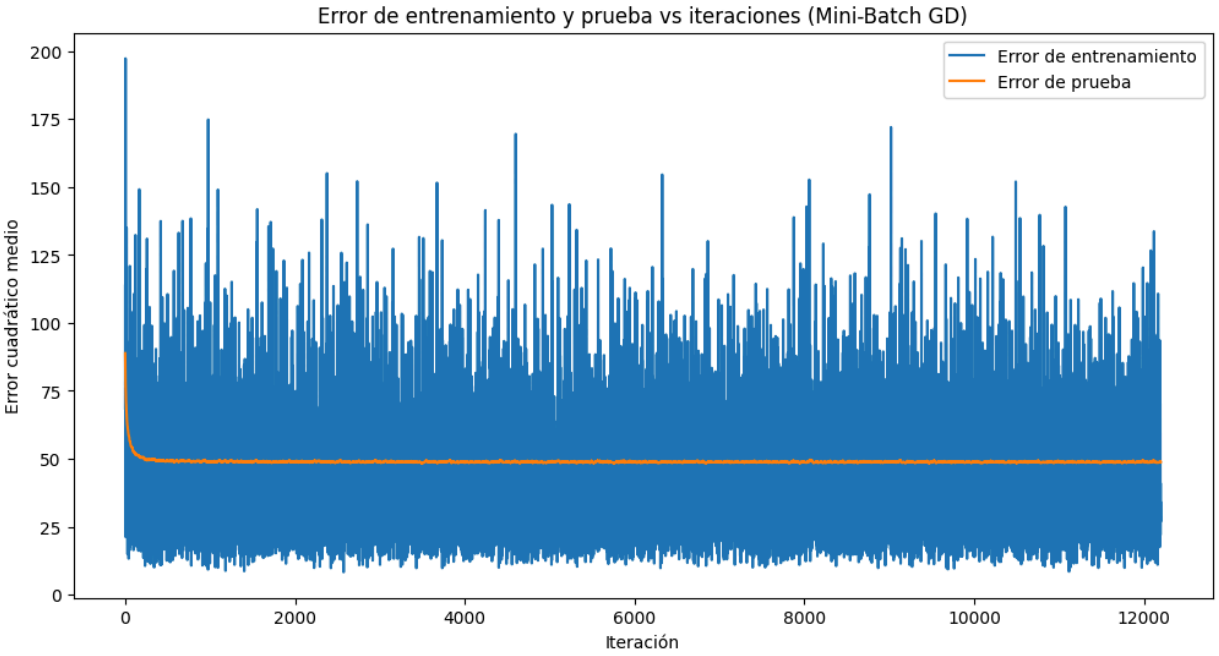
3. Iteraciones:
    - Por cada iteración:
        - Se selecciona aleatoriamente un batch de ejemplos de entrenamiento del conjunto de datos.
        - Se calcula la predicción del modelo para ese batch.
        - Se calcula el error entre la predicción y los valores reales del batch.
        - Se actualizan los coeficientes.
        - Se repiten estos pasos para cada batch hasta que se alcance el número máximo de iteraciones o se cumpla un criterio de convergencia.

4. Actualización de Coeficientes:
    - La regla de actualización de los coeficientes para el Batch Gradient Descent es similar a la del descenso de gradiente clásico, pero se aplica al batch actual. La actualización se realiza para cada coeficiente calculado para el batch actual.

5. Convergencia:
    - El proceso de iteración se repite hasta que se alcanza un número máximo de iteraciones o hasta que se cumple un criterio de convergencia, como la convergencia del error.

6. Predicción:
    - Una vez que el modelo ha sido entrenado, se pueden realizar predicciones utilizando los coeficientes ajustados.

Aplicación del modelo



MSE (entrenamiento): 40.89256279726314
MSE (prueba): 48.81808311913887

R2 Score (prueba): 0.24052823440488424

En esta aplicación del modelo las métricas de error mejoran un poco. El gráfico muestra la evolución del error a lo largo del entrenamiento y prueba.

Regresión lineal múltiple con regularización lasso

La regularización Lasso es una técnica utilizada para agregar una penalización adicional a la función de costo, con el objetivo de evitar el sobreajuste y promover la simplicidad del modelo. Se introduce una penalización proporcional al valor absoluto de los coeficientes del modelo, esta penalización logra reducir los

coeficientes más pequeños, algunos llevándolos a cero, realizando una selección de características en el modelo y así evitando el sobreajuste.

Procedimiento de Optimización

1. Minimización de la Función de Costo:
    - El objetivo es minimizar la función de costo, que ahora incluye tanto la penalización de la regresión lineal estándar (MSE) como la penalización de regularización lasso.

2. Balance entre Ajuste y Regularización:
    - El término de penalización Lasso actúa como un término de "encogimiento" que penaliza los coeficientes más pequeños y, por lo tanto, ayuda a prevenir el sobreajuste al reducir la magnitud de algunos coeficientes hacia cero.

3. Efecto de la Regularización Lasso:
    - La regularización Lasso tiene la propiedad de realizar la selección de características al hacer que algunos coeficientes sean exactamente cero evitando la inclusión de características irrelevantes.

4. Optimización con Gradiente Descendente:
    - La optimización de la función de costo con regularización Lasso generalmente se realiza utilizando técnicas como el Gradiente Descendente, que busca ajustar los coeficientes del modelo para minimizar la función de costo.

Importancia del Parámetro de Regularización:

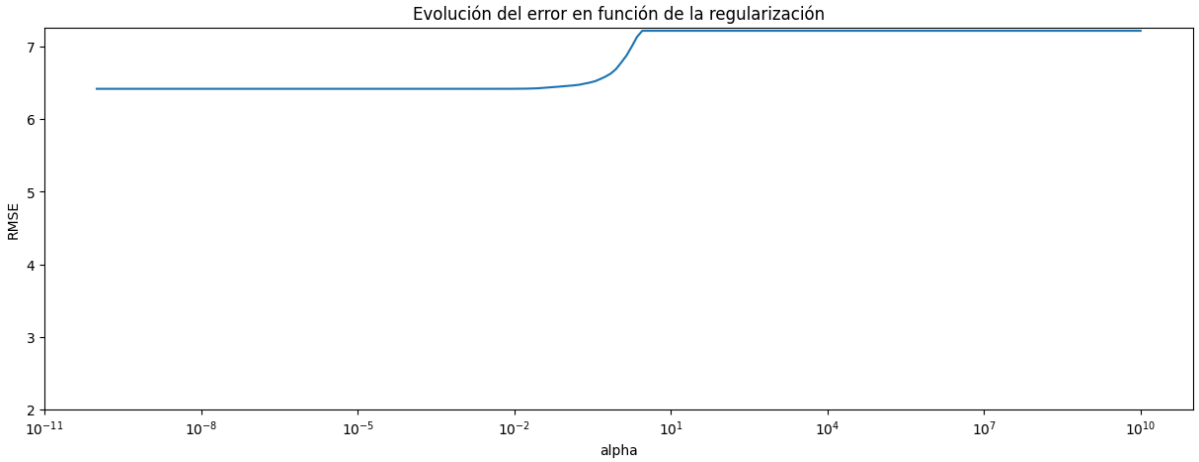
- Un λ pequeño permite que la regularización Lasso tenga un impacto menor, y el modelo tiende a comportarse más como una regresión lineal estándar.

- Un λ grande aumenta la fuerza de la regularización, lo que lleva a más coeficientes reducidos a cero y una mayor simplicidad del modelo.

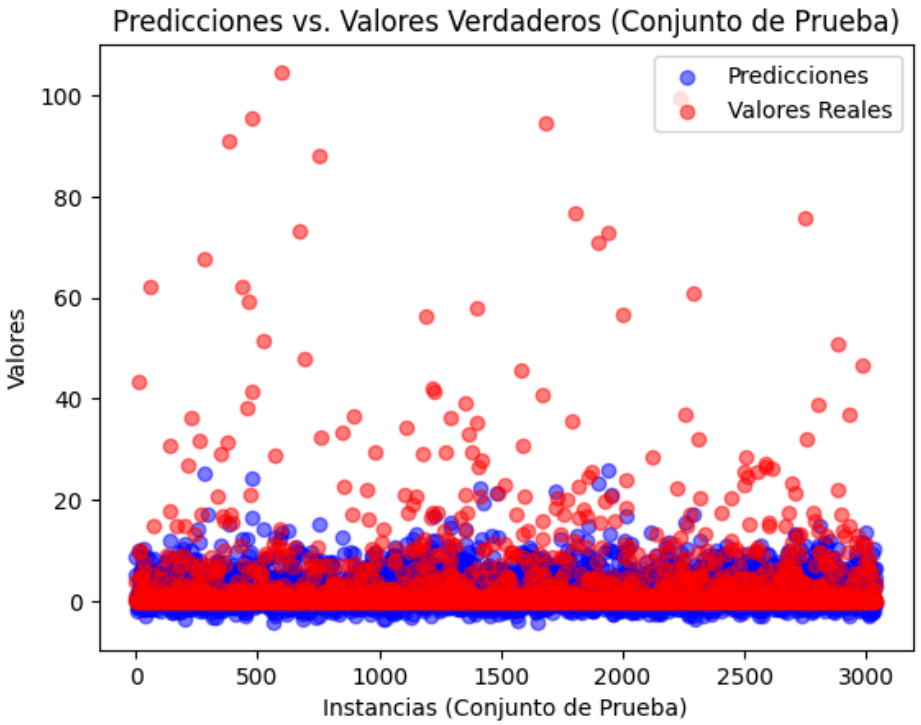
Aplicación del modelo

MSE: 48.7981071212692
R2 Score: 0.24083900462366925

En este caso las métricas MSE y R2 no se modifican mucho respecto a los modelos anteriores.



A medida que cambian los alphas no disminuye el error por lo que se debería considerar agregar más variaciones, sin embargo no creemos que se modifique demasiado el resultado.



En el gráfico de dispersión se ve que las predicciones se acumulan abajo y los casos más extremos no son predichos correctamente.

Regresión lineal múltiple con regularización ridge

La regularización Ridge al igual que la regularización lasso es una técnica utilizada para agregar una penalización adicional a la función de costo con el objetivo de evitar el sobreajuste y mejorar la estabilidad del modelo. A diferencia de la regularización Lasso, Ridge penaliza los coeficientes al cuadrado, logrando penalizar más fuertemente a los coeficientes grandes.

Aun así la regularización ridge tiene un enfoque más “armonioso” ya que reduce proporcionalmente los coeficientes tanto grande como pequeños (pero nunca llevándolos a cero como la regularización lasso).

Procedimiento de Optimización

1. Minimización de la Función de Costo:
    - El objetivo es minimizar la función de costo que ahora incluye tanto la penalización de la regresión lineal estándar (MSE) como la penalización de regularización.

2. Balance entre Ajuste y Regularización:
    - El término de penalización Ridge actúa como un término de "encogimiento" que penaliza los coeficientes más grandes y, por lo tanto, ayuda a prevenir el sobreajuste al reducir la magnitud de los coeficientes.

3. Efecto de la Regularización Ridge:
    - A diferencia de la regularización Lasso, Ridge tiende a reducir la magnitud de los coeficientes hacia cero, pero raramente los hace exactamente cero. En cambio, distribuye la "masa" de la penalización de manera más uniforme entre todos los coeficientes.

4. Optimización con Gradiente Descendente:
    - Al igual que con la regularización Lasso, la optimización de la función de costo con regularización Ridge se realiza comúnmente utilizando técnicas como el Gradiente Descendente.

Importancia del Parámetro de Regularización (λ):

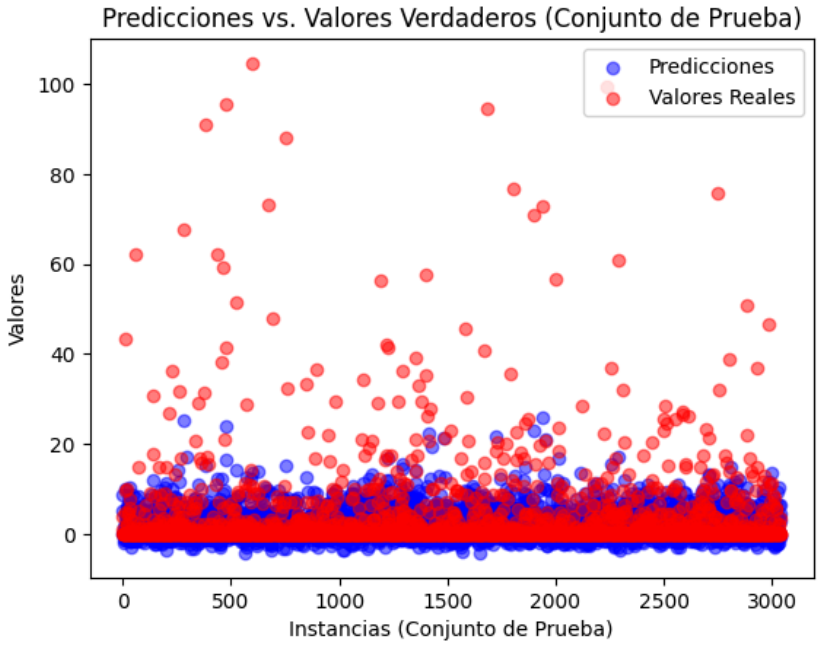
- Un λ pequeño permite que la regularización Ridge tenga un impacto menor, y el modelo tiende a comportarse más como una regresión lineal estándar.

- Un λ grande aumenta la fuerza de la regularización, lo que lleva a coeficientes más pequeños y una mayor simplicidad del modelo.

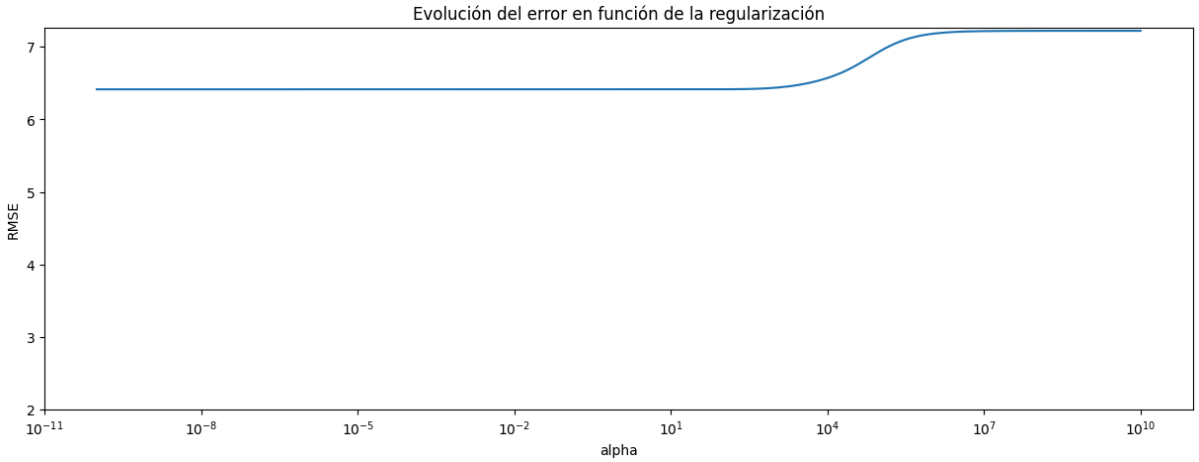
Aplicación del modelo

MSE: 48.81528483285275
R2 Score: 0.24057176785172552

En este modelo las métricas dieron ligeramente peor que en la regularización lasso. Podemos ver que el modelo sigue teniendo problemas con los casos extremos.

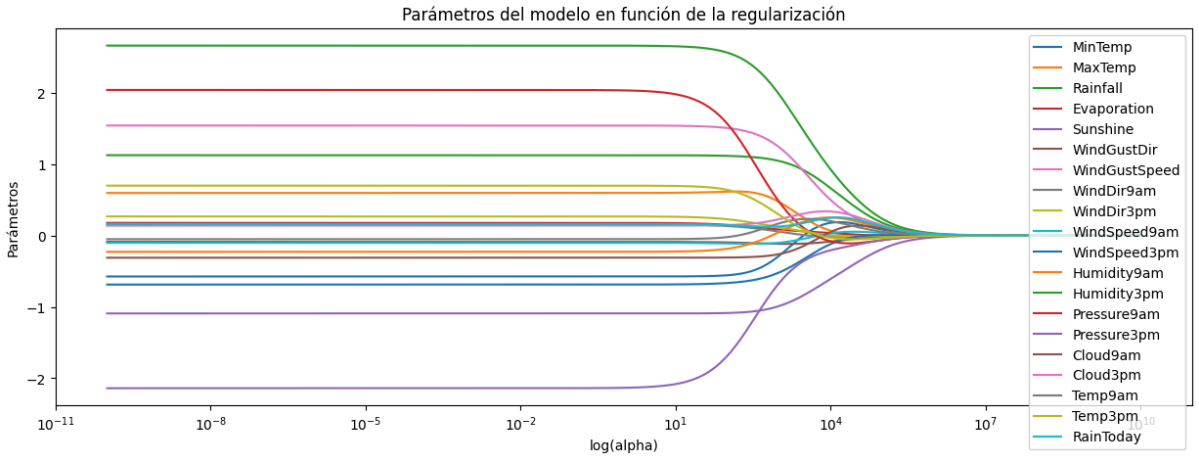


La evolución del error no disminuye según el cambio del valor alpha al igual que el modelo con regularización lasso.

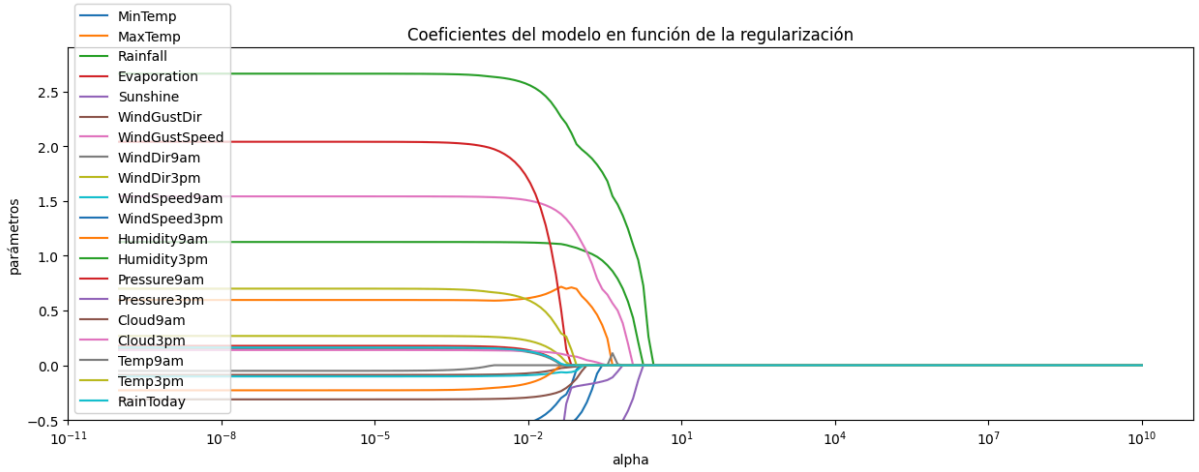


Comparación entre la evolución de los parámetros entre L1 y L2

Regularización ridge



Regularización lasso



Podemos ver reducciones de coeficientes más “suaves” en la regularización ridge y reducciones más “bruscas” en la regularización lasso, esto nos permite ver la forma en cómo se desarrollan ambas penalizaciones en función de los valores alphas.

Regresión lineal múltiple con ElasticNet

La regresión lineal múltiple con ElasticNet es una extensión de la regresión lineal que combina las penalizaciones de las regularizaciones Ridge (L2) y Lasso (L1). La función de costo de ElasticNet incluye ambos términos de penalización, permitiendo así capturar beneficios de ambas técnicas de regularización.

Procedimiento de Optimización

1. Balance Entre Ridge y Lasso:
    - ElasticNet proporciona un control más granular sobre la regularización al permitir ajustar dos parámetros (λ1 y λ2), lo que permite sintonizar la combinación exacta de Ridge y Lasso.

2. Selección de Características:
    - Similar a Lasso, ElasticNet tiene la capacidad de realizar selección de características, ya que puede anular completamente ciertos coeficientes debido a la penalización L1.

3. Manejo de Correlación de Características:
    - ElasticNet es útil cuando hay características altamente correlacionadas. La regularización L1 de Lasso tiende a seleccionar solo una característica entre características altamente correlacionadas, pero ElasticNet, al tener la penalización L2 de Ridge, puede mantener ambas características en el modelo.

4. Robustez y Estabilidad:
    - ElasticNet combina la estabilidad de Ridge con la selección de características de Lasso, lo que puede hacerlo más robusto en ciertos escenarios y menos susceptible a problemas cuando hay multicolinealidad en los datos.

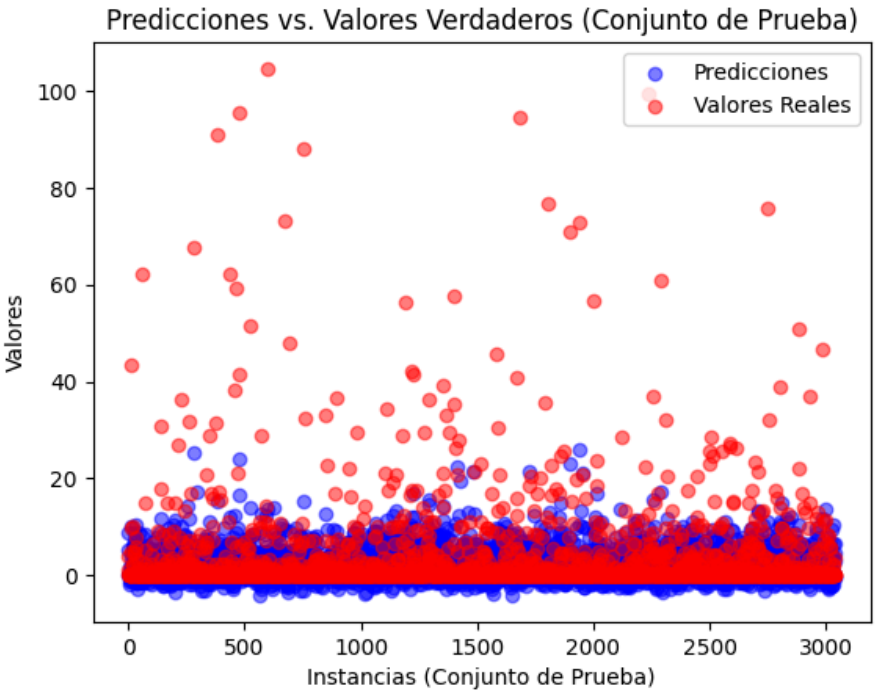
Elección de Hiperparámetros:
La elección adecuada de los parámetros de regularización (λ1 y λ2) es crucial en ElasticNet al igual que las regularizaciones anteriores. Estos hiperparámetros se ajustan a través de métodos como la validación cruzada para encontrar la combinación óptima que equilibra el ajuste del modelo y la regularización.

Aplicación del modelo

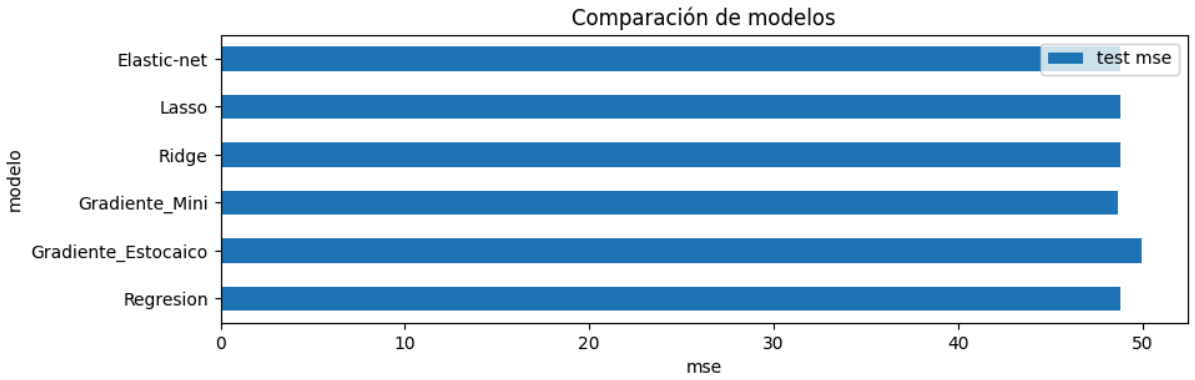
MSE: 48.82752067220799
R2 Score: 0.2403814126815731

Las métricas de ElasticNet son las peores de las tres técnicas de regularización. A pesar de haber utilizado validación cruzada para hallar los mejores hiperparámetros no logra superar los modelos anteriores respecto al rendimiento.

Viendo el gráfico de dispersión siguen siendo consistentes los problemas de predicción que han tenido el resto de modelos.



Comparación entre modelos de regresión lineal



Luego de probar distintas técnicas de regularización y optimización se realizó una comparación de modelos basándonos en la métrica de error MSE que es una

medida que cuantifica la magnitud promedio de los errores entre las predicciones de un modelo y los valores reales de un conjunto de datos.

El mejor resultado obtenido fue el modelo de regresión lineal con mini-batch dando las mejores métricas de error.

Se podrían combinar las técnicas para dar un modelo ideal que permita una métrica MSE más baja y una mayor aplicabilidad de la variabilidad de los datos con la métrica R2.

Modelos de clasificación

regresión logística

Un modelo de regresión logística es un tipo de modelo utilizado para problemas de clasificación, donde el objetivo es predecir la probabilidad de que una observación pertenezca a una categoría específica.

Este modelo utiliza la función logística (también conocida como la función sigmoide) para transformar una combinación lineal de las características del conjunto de datos en un valor entre 0 y 1. Esta transformación logística es crucial porque permite interpretar la salida como la probabilidad de pertenecer a una clase.

Funcionamiento del modelo

1. Combinación Lineal:
    - Para cada observación en el conjunto de datos, se realiza una combinación lineal de las características (variables independientes) ponderadas por ciertos coeficientes (pesos).
    - La puntuación para una observación específica se calcula como la suma ponderada de las características más un término de sesgo (intercepto).

2. Función Logística (Sigmoide):
    - La puntuación se introduce en la función logística (también conocida como función sigmoide). La función sigmoide transforma los valores a un rango entre 0 y 1.

3. Umbral de Decisión:
    - Se establece un umbral (generalmente 0.5) para determinar la clase final. Si la probabilidad calculada es mayor que el umbral, se clasifica como la clase positiva; de lo contrario, se clasifica como la clase negativa.

4. Entrenamiento del Modelo:
    - Durante el proceso de entrenamiento, se ajustan los coeficientes (pesos) mediante técnicas como el descenso de gradiente.
    - El objetivo es minimizar la diferencia entre las predicciones y las etiquetas reales utilizando una función de costo como la entropía cruzada.

5. Predicciones:
    - Una vez que el modelo está entrenado, se puede usar para hacer predicciones en nuevos datos. Dada una nueva observación, el modelo calcula la probabilidad de que pertenezca a la clase positiva y la compara con el umbral para tomar una decisión de clasificación.

Aplicación del modelo

Para evaluar los modelos de clasificación se utilizan las siguiente métricas de error:

1. Precisión (Precision):
    - Mide la precisión de las predicciones positivas realizadas por el modelo.
    - Interpretación: ¿De todas las instancias que el modelo predijo como positivas, cuántas realmente eran positivas? Una alta precisión significa que el modelo tiene una baja tasa de falsos positivos.

2. Recall (Recuperación o Sensibilidad):
    - Mide la capacidad del modelo para capturar todas las instancias positivas.
    - Interpretación: ¿De todas las instancias positivas que realmente existen, cuántas el modelo logró capturar? Un alto recall significa que el modelo tiene una baja tasa de falsos negativos.

3. F1-Score:
    - Es una métrica que combina precisión y recall en un solo número buscando un equilibrio entre ellas.

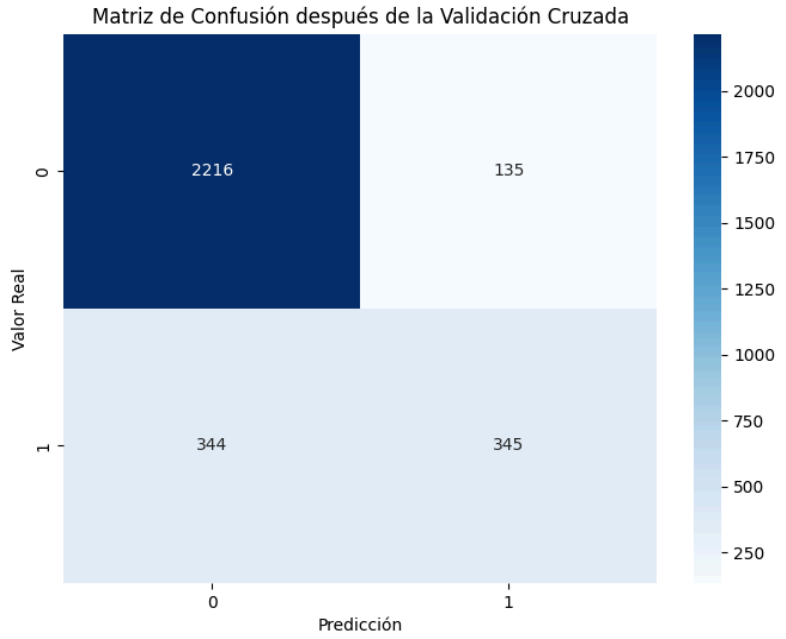
4. Exactitud (Accuracy):
    - Mide la proporción de predicciones correctas en relación con el total de predicciones.

- Interpretación: ¿Qué tan bien el modelo clasifica todas las instancias, tanto positivas como negativas? La exactitud es útil cuando las clases están equilibradas, pero puede ser engañosa en conjuntos de datos desequilibrados.

Resultados con regresión logística

	precision	recall	f1-score	support
0	0.87	0.94	0.90	2351
1	0.72	0.50	0.59	689
accuracy			0.84	3040

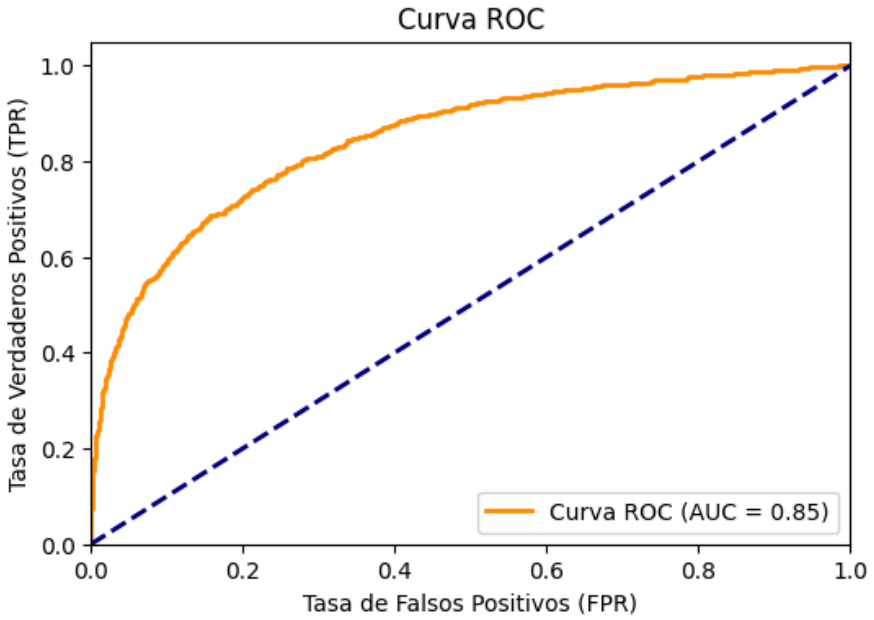
La métrica que elegimos para evaluar los modelos principalmente es el recall ya que nos indica cuántos casos reales de cada clase logra captar el modelo. Esto nos brinda información para ver cuántos casos de lluvia reales logra captar el modelo que es lo que nos interesa, las optimizaciones del modelo se hicieron para mejorar esta métrica. En nuestro caso el recall de la clase 1 es bajo, solo el 50% de los casos reales de lluvia es captado por el modelo.



La matriz de confusión nos permite ver las cantidades de aciertos y errores del modelo, nuestra intención es mejorar las predicciones de la clase 1 (casos de lluvia).

También utilizamos el gráfico de curva ROC que evalúa el rendimiento del modelo al variar los umbrales de decisión y medir la tasa de verdaderos positivos (Recall) frente a la tasa de falsos positivos (1 - Especificidad).

Este gráfico permite también obtener el área bajo la curva AUC-ROC que mide la capacidad del modelo para discriminar entre las dos clases en términos de la probabilidad predicha. Un valor AUC-ROC cercano a 1 indica que el modelo tiene una buena capacidad de discriminación entre las clases, mientras que un valor cercano a 0.5 sugiere un rendimiento similar al azar.



En este caso el rendimiento del modelo es bueno aunque podría mejorar más.

regresión logística con balance

Cuando realizamos el análisis descriptivo del conjunto de datos verificamos que las características que debemos predecir se diferenciaban en dos clases, llueve o no llueve, pero estas clases están absolutamente desbalanceadas. Para corregir esto se deben aplicar distintas técnicas de balance para que los modelos puedan clasificarlas correctamente.

Al crear el modelo de regresión logística uno de los hiperparametros que podemos agregar es class_weight, al setearlo en “balanced” estás indicando al modelo que ajuste automáticamente los pesos de las clases de manera proporcional a la inversa de sus frecuencias en el conjunto de datos.

Explicado de manera simple, si la clase A tiene menos instancias que la clase B, el modelo otorgará un mayor peso a las instancias de la clase A durante el

entrenamiento. Esto significa que los errores cometidos al clasificar la clase A se penalizarán más fuertemente que los errores en la clase B.

Procedimiento de Optimización

1. Conteo de Clases:
    - El modelo examina la distribución de las clases en el conjunto de datos.

2. Cálculo de Pesos:
    - Calcula automáticamente los pesos de clase para compensar el desequilibrio. Los pesos se calculan de manera proporcional a la inversa de la frecuencia de cada clase. Si una clase es menos frecuente, su peso será mayor.

3. Entrenamiento del Modelo:
    - Durante el entrenamiento, el modelo utiliza estos pesos para ajustar la función de costo. Los errores en la clasificación de la clase menos representada (mayor peso) tendrán un impacto más significativo en la actualización de los coeficientes del modelo.

4. Ajuste de Decision Boundary:
    - El modelo ajusta la línea de decisión (o plano, en dimensiones superiores) de manera que da más importancia a la clasificación correcta de la clase menos frecuente.

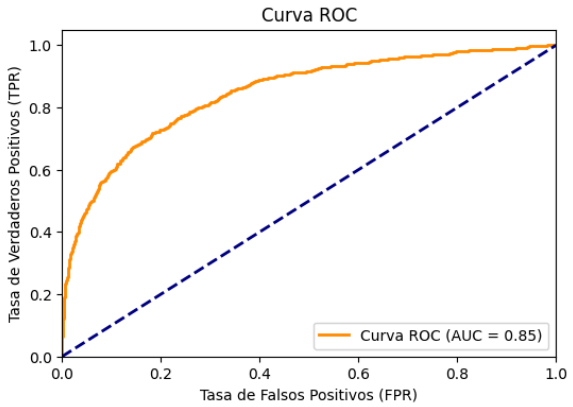
5. Predicciones Equilibradas:
    - Una vez entrenado, el modelo se puede utilizar para realizar predicciones en nuevos datos. Debido a los pesos equilibrados, se espera que el modelo tenga un mejor rendimiento en la clasificación de instancias de clases menos representadas.

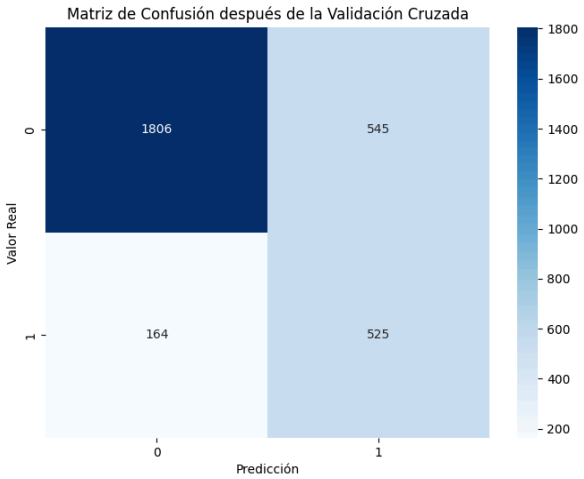
Aplicación del modelo

Resultados con regresión logística con balance

	precision	recall	f1-score	support
0	0.92	0.77	0.84	2351
1	0.49	0.76	0.60	689
accuracy			0.77	3040

Se logró mejorar la métrica recall de la clase de lluvia, el balance funcionó muy bien, ahora el modelo logra captar el 76% de los casos reales de lluvia.





Empeora un poco la cantidad de casos correctamente clasificados de la clase 0 pero ambos recalls son casi iguales ya que reconoce el 77% de los casos en los que no llueve. Esto consideramos que está bien ya que está más balanceado el reconocimiento de casos reales por parte del modelo en ambas clases.

En tanto al gráfico de curva ROC y área bajo la curva AUC-ROC mejora ligeramente con respecto al modelo anterior.

regresión logística con balance SMOTE

La técnica SMOTE es utilizada para generar de manera sintética instancias adicionales de la clase minoritaria para igualar su representación en comparación con la clase mayoritaria.

Procedimiento de Optimización

1. Identificación de Vecinos:
    - Para cada instancia en la clase minoritaria, se calculan los k vecinos más cercanos en el espacio de características.

2. Selección de Vecinos:
    - Se elige al menos uno de esos vecinos cercanos y se etiqueta como el "vecino seleccionado".

3. Generación Sintética:
    - Se crea una nueva instancia sintética tomando una combinación convexa de las características de la instancia original y el vecino seleccionado. Esta nueva instancia se coloca en el espacio de características entre la instancia original y el vecino.

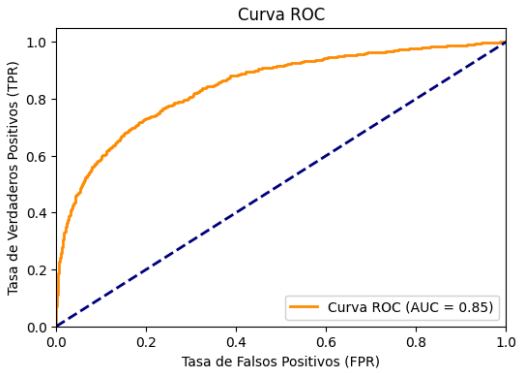
4. Repetición del Proceso:
    - Estos pasos se repiten hasta que se ha alcanzado el nivel deseado de equilibrio entre las clases o hasta que el conjunto de datos alcanza el tamaño deseado.

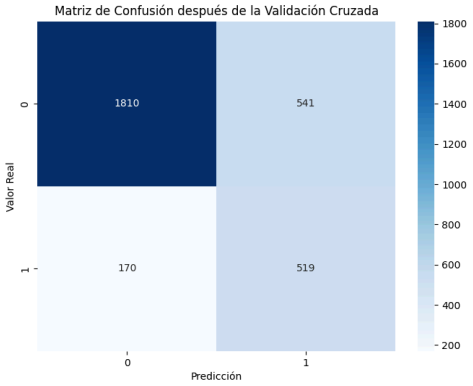
Aplicación del modelo

Resultados regresión logística con SMOTE

	precision	recall	f1-score	support
0	0.91	0.77	0.84	2351
1	0.49	0.75	0.59	689
accuracy			0.77	3040

El recall y precisión de la clase 1 disminuye ligeramente. La técnica no es muy efectiva ya que no presenta mejoras significativas respecto del modelo anterior.





regresión logística con balance oversampling

Al igual que la técnica anterior implica aumentar la cantidad de instancias de la clase minoritaria para equilibrar la distribución de clases en un conjunto de datos. Sin embargo la diferencia está en que esta técnica simplemente duplica instancias existentes de la clase minoritaria para aumentar su representación en el conjunto de datos.

Procedimiento de Optimización

1. Identificación de la Clase Minoritaria:
    - Se identifica la clase minoritaria en el conjunto de datos, es decir, la clase que tiene menos instancias.
2. Selección de Instancias a Duplicar:
    - Se eligen aleatoriamente instancias de la clase minoritaria para duplicar. La cantidad de instancias seleccionadas puede depender de la diferencia en tamaño entre las clases.
3. Duplicación de Instancias:
    - Las instancias seleccionadas se duplican, creando copias adicionales de las mismas. Esto se puede hacer para igualar el número de instancias de la clase mayoritaria o para alcanzar un equilibrio deseado.
4. Inclusión en el Conjunto de Datos:
    - Las instancias duplicadas se agregan al conjunto de datos original, aumentando así la representación de la clase minoritaria.

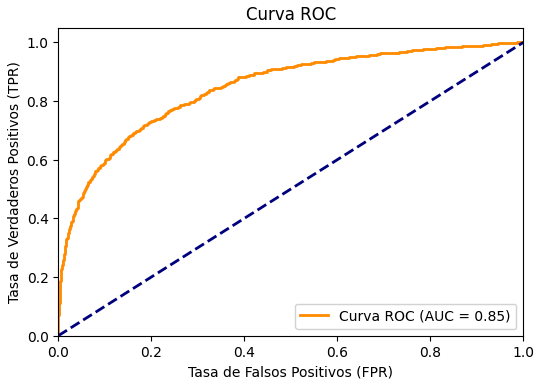
Aplicación del modelo

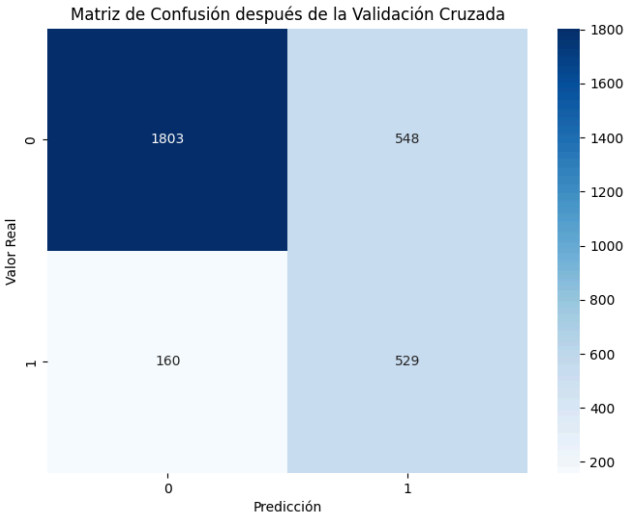
Resultados regresión logística con Oversampling

	precision	recall	f1-score	support
0	0.92	0.77	0.84	2351
1	0.49	0.77	0.60	689
accuracy			0.77	3040

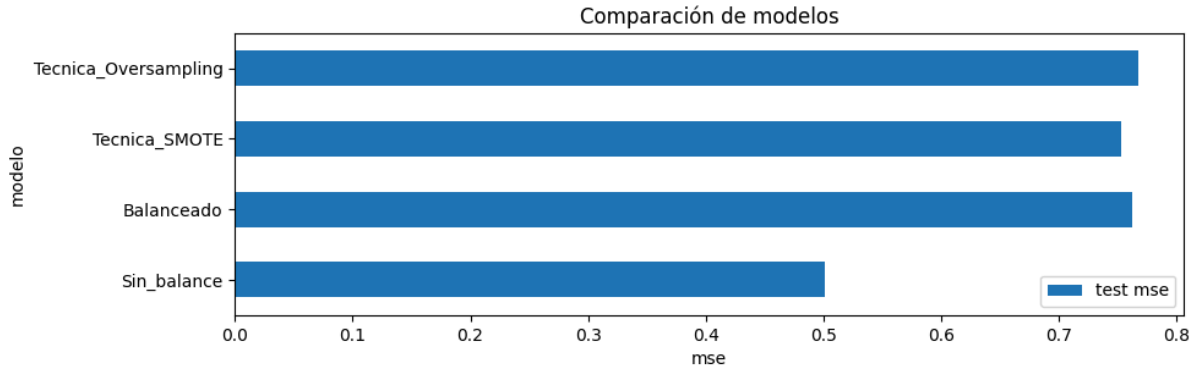
En este caso la técnica de balance funciona mejor que la anterior, logrando igualar los recalls de ambas clases y aumentando el recall de la clase 1 aunque sacrificando un poco la precisión.

Los casos bien clasificados son bastantes en ambas clases y el rendimiento del modelo mejoró a pesar del desbalance inicial.





Comparación entre modelos de regresión logística



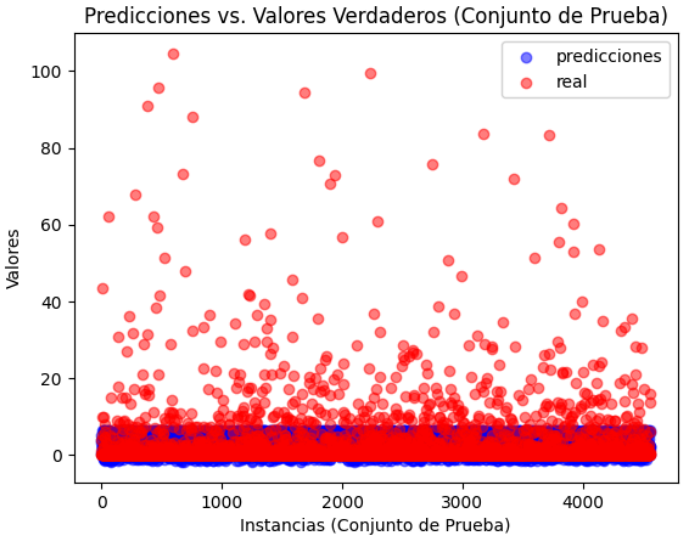
Se guardaron los valores de recalls de los distintos modelos y se comparan entre sí, dando como mejor modelo al que usa la técnica Oversampling, ya que tiene un recall de la clase 1 (la de lluvia) alto y logra predecir un porcentaje grande de los días que llueven. Las técnicas de balanceo mejoraron significativamente el rendimiento del modelo.

Modelos base

Los modelos base son modelos simples y fundamentales que se utilizan como punto de partida o referencia en un análisis o tarea específica. Estos modelos proporcionan un punto de comparación para evaluar el rendimiento de modelos más complejos y sofisticados.

Modelo base de regresión

MSE: 53.890473154448244
R2: 0.08484286882000747



Este modelo base se entrenó con una sola característica para que sea un modelo simple, la métrica MSE es un poco más alta con respecto a los modelos anteriores, mientras que el R2 es bajísimo en comparación con el resto de modelos de regresión.

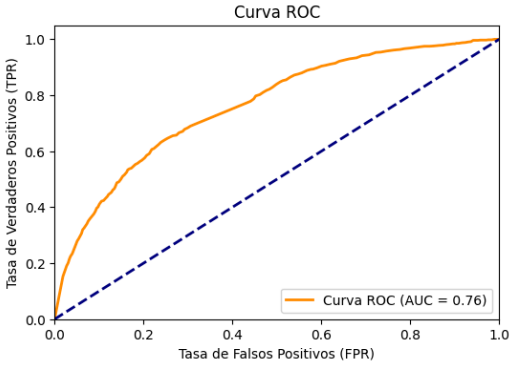
Los modelos anteriores logran obtener mejores métricas que este modelo base.

Modelo base de clasificación

matriz de confusión:
[[3238 280]
[668 374]]

Classification Report

	precision	recall	f1-score	support
0	0.83	0.92	0.87	3518
1	0.57	0.36	0.44	1042
accuracy			0.79	4560



En el modelo base de clasificación al igual que en el de regresión se utiliza una sola característica para entrenar al modelo y que logre predecir las clases. vemos que tiene un recall bajísimo de la clase 1 mientras que la clase 0 es muy alto, esto es por el desbalance que hay entre clases.

Los modelos anteriores logran obtener mejores métricas que este modelo base.

Redes neuronales

Las redes neuronales son modelos computacionales inspirados en la estructura y función del cerebro humano. En su esencia, estas redes consisten en "neuronas artificiales" interconectadas, organizadas en capas. Cada neurona recibe entradas, realiza cálculos ponderados y aplica una función de activación para producir una salida.

Estructura y funcionalidad de redes neuronales

1. Neuronas Artificiales:
    - En una red neuronal, las "neuronas artificiales" son unidades básicas de procesamiento. Cada neurona toma varias entradas, realiza cálculos ponderados y aplica una función de activación para producir una salida.

2. Capas:
    - Las neuronas se organizan en capas. Una red neuronal típica consta de tres tipos de capas: la capa de entrada, que recibe datos; una o más capas ocultas, que realizan cálculos intermedios; y la capa de salida, que produce la salida final.

3. Conexiones y Pesos:
    - Las conexiones entre neuronas tienen pesos asociados. Estos pesos determinan la importancia de cada conexión. Durante el entrenamiento, estos pesos se ajustan para que la red aprenda patrones y relaciones en los datos.

4. Funciones de Activación:
    - Después de calcular la suma ponderada de las entradas, cada neurona aplica una función de activación. Esta función introduce no linealidades en la red, permitiendo que la red aprenda y represente patrones complejos.

5. Propagación hacia Adelante:
    - La información fluye a través de la red desde la capa de entrada hasta la capa de salida. Cada neurona realiza sus cálculos y pasa la información a la siguiente capa.

6. Retropropagación del Error:
    - Durante el entrenamiento, se compara la salida predicha con la salida real para calcular el error. Luego, este error se retropropaga a través de la red para ajustar los pesos y mejorar el rendimiento.

7. Aprendizaje:
    - A medida que la red se expone a más datos durante el entrenamiento, ajusta sus pesos para reducir el error y mejorar su capacidad para realizar tareas específicas.

Red neuronal de clasificación

Arquitectura de la red

La red neuronal que se utilizó tiene dos capas ocultas de diez neuronas cada una y una capa de salida con una neurona.

- capas ocultas:
    - Funcion de activacion ReLu:
      ReLU es una función de activación que se aplica a la salida de cada neurona para introducir no linealidades en el modelo su función es f(x)=max(0,x). Esto significa que si la entrada x es positiva, la salida es x; si es negativa, la salida es cero.

- Capa de salida:
    - Función de activación sigmoidea:
      La capa de salida tiene 1 neurona con una activación sigmoide, comúnmente utilizada en problemas de clasificación binaria.

Compilación del modelo

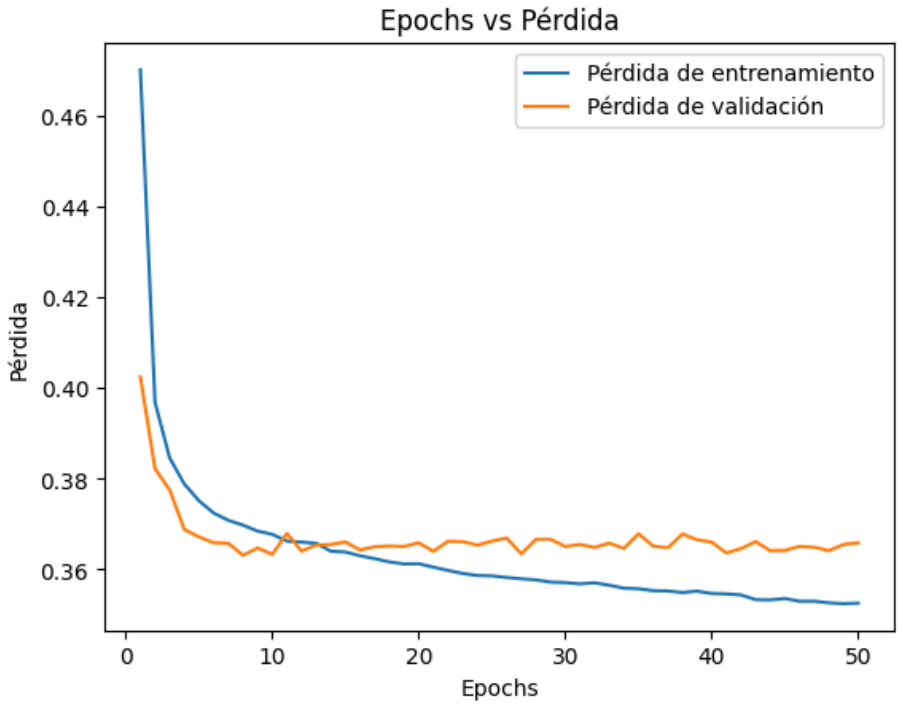
Función de pérdida
En la compilación del modelo se utiliza entropía binaria cruzada, es común para problemas de clasificación binaria. Mide la discrepancia entre las probabilidades predichas y las etiquetas reales.

Función de optimización
Como función de optimización se utiliza la función adam que combina la idea del momento en el Gradiente Descendente con la adaptación de la tasa de aprendizaje en RMSProp. Utiliza dos momentos, uno para el gradiente y otro para el cuadrado del gradiente, adaptando la tasa de aprendizaje para cada parámetro haciéndolo un algoritmo adaptativo y robusto.

Resultados
Se aplicó el modelo con un número total de 50 épocas obteniendo los siguientes resultados:
Accuracy: 0.7733552631578947
Precision: 0.5980783630540166
Recall: 0.7733552631578947

F1-score: 0.6745161282449307
Se obtuvo un recall alto siendo este el mejor modelo obtenido en clasificación.

También se puede visualizar la reducción de la pérdida a medida que avanzan las épocas en entrenamiento y validación. Esto nos muestra que el modelo no sufrió sobreajuste.



Red neuronal de regresión

Arquitectura de la red

La red neuronal que se utilizó tiene dos capas ocultas de diez neuronas cada una y una capa de salida con una neurona.

- capas ocultas:
  - Funcion de activacion ReLu

- Capa de salida:
  - En el caso de regresión no se utiliza funcion de activacion en la capa de salida

Compilación del modelo

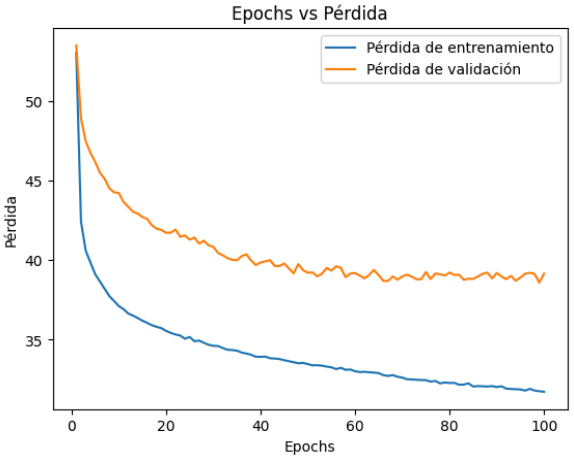
Función de pérdida
Como función de pérdida se utilizó MSE que mide el promedio de los cuadrados de las diferencias entre las predicciones del modelo y los valores reales. Esta función penaliza de manera significativa las grandes discrepancias entre las predicciones y los valores reales.

Función de optimización
Como función de optimización se utiliza la función adam al igual que en las redes neuronales para clasificación.

Resultados
Se aplicó el modelo con un número total de 100 épocas obteniendo los siguientes resultados:

Mean Squared Error (MSE): 69.86009210526315

En este modelo el error es mucho mayor respecto a los modelos anteriores, incluso aumentando las épocas no dio mejores resultados. Quizás cambiando la estructura de la red haciéndola mucho más grande se podrían llegar a obtener resultados más óptimos, sin embargo se requiere un alto costo computacional.



Búsqueda de hiperparametros

Se utilizó optuna que es una biblioteca de optimización de hiper parámetros para Python que se utiliza para encontrar los mejores valores de un modelo de aprendizaje automático. Su enfoque está basado en la optimización de funciones objetivas, las cuales serían la función de pérdida o la métrica de rendimiento que se busca maximizar o minimizar.

Decidimos utilizarlo porque tiene un enfoque de optimización bayesiana para encontrar los mejores hiperparámetros. Esto implica construir un modelo de regresión probabilística que estima la relación entre los hiperparámetros y la función objetivo, y luego utiliza esta información para guiar la búsqueda hacia regiones prometedoras del espacio de búsqueda.

Luego de haberlo aplicado no se obtuvieron resultados diferentes a los anteriores, quizás extendiendo el rango de búsqueda se podrían mejorar sin embargo el costo computacional es muy alto.

MLOPS

MLOPS busca integrar y optimizar el ciclo de vida completo de desarrollo y operación de modelos de machine learning. Combina prácticas de desarrollo de software, administración de sistemas y principios de operaciones para garantizar la implementación efectiva, la monitorización continua y la iteración eficiente de modelos en entornos de producción.

Primer Nivel de MLOPS: Automatización del Despliegue de Modelos

- En este nivel, se prioriza la automatización del despliegue de modelos en entornos de producción. Esto implica establecer flujos de trabajo automáticos para llevar los modelos desde el desarrollo hasta la implementación. Algunos aspectos clave incluyen:

    - Gestión de Versiones: Controlar las versiones de los modelos para mantener un historial claro y permitir cambios controlados.

    - Automatización del Despliegue: Desarrollar procesos automatizados para poner en producción nuevos modelos y actualizar versiones existentes. Esto puede incluir la orquestación de contenedores, la gestión de dependencias y la configuración de infraestructuras necesarias.

    - Integración con Sistemas Existentes: Asegurarse de que los modelos se integren sin problemas con los sistemas existentes de la organización. Esto implica considerar aspectos como la compatibilidad de datos, la seguridad y la escalabilidad.

En nuestro caso utilizamos la librería Streamlit de Python que fue diseñada para simplificar la creación de aplicaciones web interactivas con poco esfuerzo.

Con esta herramienta logramos fácilmente convertir nuestro código en una aplicación web funcional sin necesidad de conocimientos avanzados de desarrollo web permitiendo así la interacción de los usuarios con los mejores modelos seleccionados.

Los modelos seleccionados como mejores son mini batch (regresión) y para redes neuronales (clasificación). La elección la hicimos teniendo en cuenta las métricas de

error para cada modelo, MSE (principal) y R2 (secundaria) para modelos de regresión, recall (principal) y f1-score (secundaria) para modelos de clasificación. Sin embargo el deploy se realizó solo con los modelos de redes neuronales ya que era más sencillo.

Conclusiones

En este proyecto, se aplicaron diversas técnicas, como optimización, regularización, análisis y balance, para mejorar la implementación de modelos. Los análisis realizados en el dataset proporcionaron información crucial para tomar decisiones respecto a las características y conjunto de datos que se utilizarían en los modelos.

Durante la implementación, se enfrentaron varios desafíos que se abordaron mediante prueba y error, aplicando diferentes técnicas para favorecer el entrenamiento y prueba de los modelos. El desbalance entre las clases de datos, especialmente notable en la escasez de datos de lluvia, representó uno de los mayores obstáculos. Este problema se abordó de manera más efectiva en los modelos de clasificación en comparación con los modelos de regresión.

La introducción de técnicas avanzadas, como redes neuronales, el uso de SHAP para explicabilidad y estrategias robustas de división de datos, contribuyeron a mejorar la interpretación de los datos y los resultados. Sin embargo, debido a limitaciones computacionales, las redes neuronales se mantuvieron en versiones más simples, y se reconoce que con más recursos computacionales, los resultados en regresión podrían haber sido aún más óptimos.

Finalmente, la implementación de los mejores modelos con prácticas de MLOPS proporcionó una valiosa visión sobre cómo trabajar con implementaciones de modelos a gran escala. A pesar de las limitaciones, este despliegue permitió comprender aspectos prácticos relacionados con la gestión y mantenimiento de modelos en entornos de producción.