Análisis y Reporte sobre el desempeño del modelo

INTELIGENCIA ARTIFICIAL AVANZADA PARA LA CIENCIA DE DATOS

Osvaldo Del Valle Mejía

A01275702

11 - Septiembre - 2024

Índice

Introducción **3**
Procedimiento **4**
ETL 4
Regresión Lineal 5
Implementación manual 5
Función Hipótesis 6
Función de error cuadrático medio (MSE) 7
Función de descenso por gradiente 7
Conjuntos del dataset 8
Evaluación del Modelo **9**
Regresión lineal con framework **11**
Mejoría del ETL 11
Random forest **12**
Implementación 12
Grid search CV y Bagging 13
Resultados obtenidos 14
Comparación de resultados 16
Comparación de resultados predichos contra esperados con implementación a mano 17
Comparación de resultados predichos contra esperados con uso de framework 17
Comparación de resultados predichos contra esperados con uso de random forest con overfitting y una alta variabilidad 18
Comparación de resultados predichos contra esperados con uso de random forest con overfitting solucionado parcialmente con bagging y probar con diferentes hiperparametros 18
Referencias **19**

Introducción

Este documento presenta los detalles del reto "Módulo 2 Análisis y Reporte sobre el desempeño del modelo", con un enfoque en la modelación de regresión lineal para predecir el alquiler de bicicletas públicas por lo tanto se decidió el uso del dataset del sistema de compartición de bicicletas de Seúl, que contiene información detallada sobre el alquiler de bicicletas por hora, junto con datos meteorológicos e información sobre días festivos. El dataset cuenta con alrededor de 8000 instancias con cada una registrando datos horarios de alquiler y 13 características que incluyen temperatura, humedad, velocidad del viento, visibilidad, punto de rocío, radiación solar, nevadas, lluvias y datos adicionales sobre el alquiler de bicicletas, sin embargo hay que considerar que este dataset es enfocado en la predicción de estos datos para conocer si es un día laboral, pero considerando que un enfoque más interesante seria el predecir la cantidad de bicis que serían necesarias para un dia en especifico en base a los parámetros aportados se decidió este enfoque.

Por lo tanto el objetivo principal es entender cómo estas características influyen en las predicciones de alquiler y cómo se comparan con los recuentos reales de alquiler, este dataset está enfocado a un modelo de regresión a predecir la cantidad de bicicletas en base a los features seleccionados para el entrenamiento lo que se puede utilizar para generar datos y conocer cuántas bicicletas serían necesarias para asegurar una disponibilidad óptima, para que en algún contexto donde se aplicará el modelo se pudiera minimizar los tiempos de espera y garantizando un suministro constante.

Procedimiento

En este proceso se emplea una técnica de regresión lineal implementando de manera manual además de una implementación apoyada de un framework para utilizar una solución ya construida y probada, además se opta por otra opción de modelado para utilizar random forest como modelo principal asistido por un modelo de bagging para mejorar el resultado debido al overfitting, varianza y valor de r2 con el fin de predecir la cantidad de bicicletas alquiladas en el sistema de compartición de bicicletas de Seúl. A continuación, se detalla la teoría usada y los métodos implementados en el código.

ETL

El proceso de ETL significa extract, transform, load, este es un proceso muy importante en la gestión de datos para integrar y preparar información que proviene de diversas fuentes, este proceso además se utiliza para preparar los datos para enviar a un lugar donde se pueden emplear para situaciones variadas donde se pueden analizar y utilizar de manera efectiva, la fase de extracción implica obtener datos de diversas fuentes, como bases de datos, archivos o sistemas en la nube, en la fase de transformación los datos se limpian, agregan y reorganizan para que sean útiles y consistentes eliminando errores y duplicados, finalmente en la fase de carga los datos transformados se almacenan en un sistema centralizado como un data warehouse donde pueden ser accedidos y utilizados por usuarios finales o aplicaciones de inteligencia artificial.

En el análisis de datos realizado se utilizaron varias variables clave extraídas del conjunto de datos original para construir el modelo predictivo, las variables incluyeron 'Date' que se transformó para capturar únicamente el mes en lugar de la fecha completa ya que se pudiera encontrar algun patron en base al mes, se considero que el año no seria importante ya que este seria mas para datos historicos y no se cree que resulte tan significativo en cuanto a aportacion al modelo, ademas el numero del dia tampoco se considera significativo ya que este no podria representar alguna tendencia o algo similar siendo que el numero ocurre cada mes por lo tanto no es tan probable

que genere algun patron, entonces se elimino la columna original para centrarse en posibles relaciones entre los meses y el uso de bicicletas, por otro lado las variables seleccionadas fueron: 'rentedBikeCount' como la variable objetivo a predecir, 'hour' para capturar el patrón de uso a lo largo del día, 'temperature', 'humidity', 'windSpeed', 'visibility', 'dewPoint', 'solarRadiation', 'rainfall', y 'snowfall', que representan factores ambientales clave que influyen en la demanda de bicicletas, además la columna 'season' fue transformada utilizando el método One-Hot Encoding para convertirla en variables dummies que faciliten su uso en modelos de machine learning, además, la columna 'holiday' se transformó de un formato categórico a valores numéricos para que el modelo pudiera interpretarla correctamente y finalmente la variable 'functioningDay', que inicialmente se usaba para indicar si era un día hábil o no, fue eliminada ya que todos sus valores eran iguales, por lo que no aportaba información relevante al modelo, con este procesamiento se tiene un manejo del dataset orientado a lo que se necesita para manejarlo

Regresión Lineal

La regresión lineal es una técnica estadística fundamental en el aprendizaje supervisado, utilizada para modelar la relación entre una variable dependiente y una o más variables independientes. En este caso, el objetivo es predecir el número de bicicletas alquiladas (variable dependiente) en función de varias características meteorológicas y temporales (variables independientes).

Implementación manual

Para la implementación manual se decidió utilizar únicamente las variables independientes de ‘Hour’, ‘Temperature’ y ‘Season’, esto basa en un criterio propio ya que estas podrían representar mayores patrones en la cantidad de bicis rentadas, por ejemplo si es que hacia mucho mas calor en un dia se podria pensar que la cantidad de bicis disminuye o bien si es una hora de la madrugada lo mismo podría ocurrir.
En esta implementación los valores de ‘Season’ se convirtieron a valores numéricos consecutivos.

La implementación incluye funciones como la función de hipótesis que se encarga de calcular la predicción del modelo utilizando los parámetros actuales y las características del conjunto de datos, la función mean square error para evaluar el rendimiento del modelo calculando el error cuadrático medio, la función de gradiente descendiente para ajustar los parámetros del modelo para minimizar el error, además, la función de escalado que sirve para normalizar las características y así mejorar la convergencia del modelo. También en la implementación se incluye una función para dividir el conjunto de datos en conjuntos de entrenamiento, validación y prueba, así como para calcular el valor R cuadrado que mide la capacidad del modelo para explicar la variabilidad de los datos. Finalmente se generan gráficos que muestran el error de entrenamiento y validación a lo largo de las épocas y comparan los valores reales con los predichos en los conjuntos de entrenamiento y prueba

Función Hipótesis

La función hipótesis es fundamental en el modelo de regresión lineal, ya que determina cómo se generan las predicciones basadas en los parámetros del modelo y las características de entrada. En esencia la función de hipótesis toma los parámetros actuales del modelo, que se ajustan durante el proceso de entrenamiento, y los aplica a los datos de entrada para obtener una estimación de la variable que se desea predecir. La precisión de estas predicciones depende de la calidad de los parámetros y de cómo estos capturan la relación entre las características de entrada y la variable objetivo. En este análisis, la función hipótesis calcula la cantidad esperada de bicicletas alquiladas para cada conjunto de características de entrada proporcionando así una base para evaluar el desempeño del modelo. Esta función funciona en base al cálculo de una multiplicación matricial entre las características de entrada y los parámetros, lo que permite estimar la variable objetivo, por lo tanto se puede aplicar a la implementación ya que se busca que en base a las variables de entrada se busquen coeficientes con los que se llegue a un resultado de la variable dependiente en este caso la cantidad de bicis a rentar.

Función de error cuadrático medio (MSE)

El error cuadrático medio (MSE) es una métrica para evaluar la precisión del modelo de regresión, esta medida calcula la diferencia entre las predicciones realizadas por el modelo y los valores reales observados en los datos, entonces el MSE ofrece una indicación de cuánto se desvían las predicciones del modelo en promedio respecto a los valores reales. En el modelo un MSE bajo indica que el modelo está haciendo predicciones más precisas mientras que un MSE alto indica que las predicciones están más alejadas de los valores reales. Esta métrica es esencial para ajustar y mejorar el modelo, ya que proporciona una señal clara sobre la efectividad del modelo en representar la realidad.

En mi implementación, la función mean square error calcula el MSE tomando como parámetros los coeficientes actuales del modelo (‘Hour’, ‘Temperature’, ‘Season’), las características del conjunto de datos (xFeatures) y los valores reales observados (yResults). La función comienza calculando las predicciones del modelo multiplicando las características por los parámetros (hipótesis del modelo). A continuación, se calcula el error, que es la diferencia entre las predicciones y los valores reales. Este error se eleva al cuadrado para penalizar los errores grandes de manera más significativa y luego se promedia sobre el número total de observaciones.

El MSE se utiliza en cada iteración del gradiente descendente para evaluar el rendimiento del modelo y guiar el ajuste de los parámetros, con un valor de MSE bajo durante el entrenamiento y la validación indica que el modelo está mejorando y que sus predicciones están más cerca de los valores reales

Función de descenso por gradiente

El gradiente descendiente es un método de optimización ampliamente utilizado en el aprendizaje automático para ajustar los parámetros del modelo con el fin de minimizar el error, este algoritmo trabaja iterativamente, ajustando los parámetros en pequeños pasos con el objetivo de reducir el error del modelo. El gradiente descendiente es un método esencial en la optimización de modelos de aprendizaje automático y en esta

implementación se utiliza para ajustar los parámetros del modelo de regresión lineal y minimizar el error cuadrático medio (MSE), la función gradiente descendiente evalúa iterativamente los parámetros del modelo, ajustándose en pequeños pasos controlados por una tasa de aprendizaje, para que en cada iteración se calcule la diferencia entre las predicciones actuales y los valores reales para que a partir de esta diferencia se usa para calcular el gradiente, que indica en qué dirección y magnitud deben cambiar los parámetros para reducir el error. Este proceso continúa a lo largo de varias épocas hasta que los parámetros convergen a valores que minimizan el error de manera significativa. En esta implementación el gradiente descendiente es importante para mejorar la precisión del modelo al ajustar dinámicamente los parámetros para optimizar la predicción del número de bicicletas alquiladas, por lo que gracias a este proceso iterativo el modelo va afinando su capacidad para capturar las relaciones entre las variables independientes y la variable objetivo.

Conjuntos del dataset

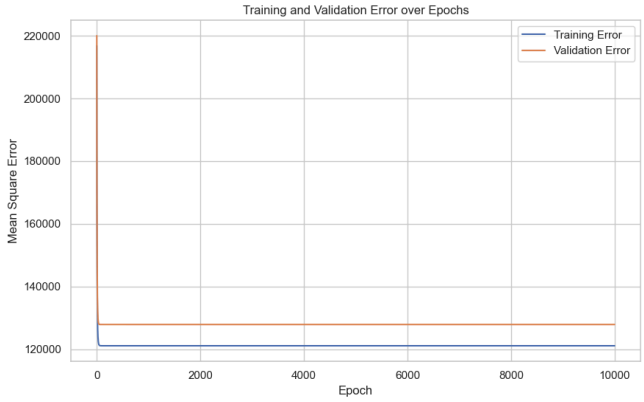
En el proceso de construcción y evaluación del modelo de regresión lineal, la adecuada división de los datos en conjuntos de entrenamiento, validación y prueba ha sido muy importante para asegurar un modelo confiable. En mi implementación el conjunto de entrenamiento representó el 60% de los datos totales, permitiendo que el modelo ajustara sus parámetros mediante el algoritmo de gradiente descendiente con suficiente información. El conjunto de validación que también constituyó el 20% de los datos, fue necesario para evaluar el rendimiento del modelo durante el entrenamiento. Esta evaluación ayudó a identificar y corregir problemas como el sobreajuste y optimizar los parámetros.

Finalmente, el conjunto de prueba que representó el 20% restante de los datos, permitió evaluar la capacidad del modelo para generalizar y hacer predicciones precisas sobre datos nuevos que no se habían utilizado durante el entrenamiento.

Evaluación del Modelo

La evaluación del modelo se realiza utilizando dos métricas principales:

- **Error cuadrático medio (MSE)**: Mide el rendimiento del modelo tanto en el conjunto de entrenamiento como en el de validación.

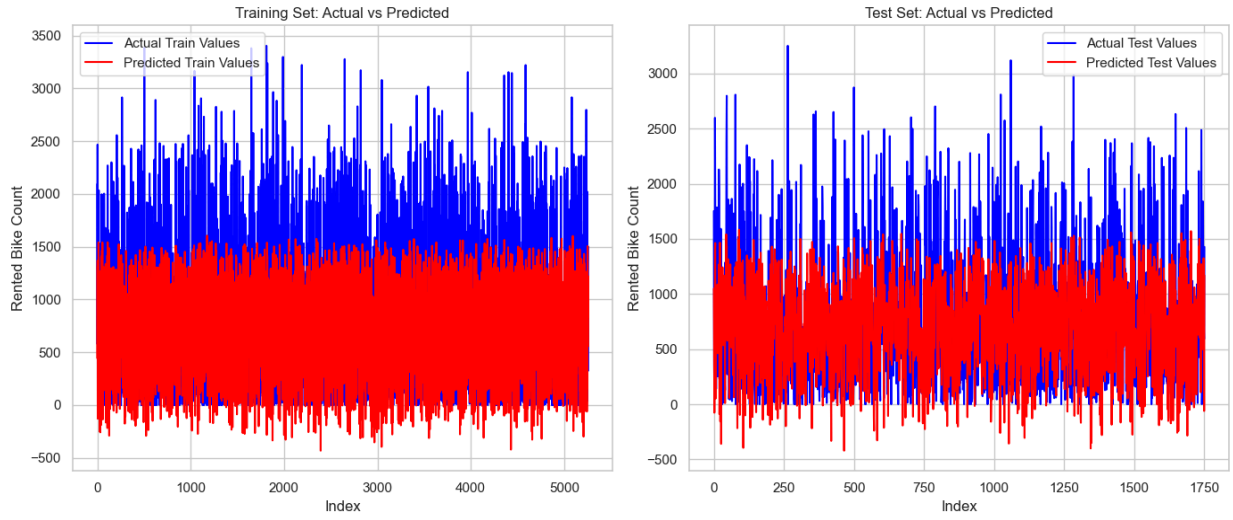


- **R-Cuadrado (R^2)**: Indica la proporción de la varianza en la variable dependiente que es explicada por las características del modelo

```
0:      Training Error = 216845.7461573594      Validation Error = 220067.26600377675
500:    Training Error = 121070.45928986992     Validation Error = 127858.19172847256
1000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
1500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
2000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
2500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
3000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
3500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
4000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
4500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
5000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
5500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
6000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
6500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
7000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
7500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
8000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
8500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
9000:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
9500:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
9999:   Training Error = 121070.45928986996     Validation Error = 127858.19172811788
Final Parameters:[ 704.52663623  784.71269729 1016.4542065   -33.8598344 ]
You can find graphs with result in results folder.
R squared for Training Set: 0.41879626117621094
R squared for Validation Set: 0.3989733102352303
```

En base a los datos que se obtuvieron al evaluar el modelo, tanto en los valores impresos cada algunas épocas con respecto al error en entrenamiento y en validación así como con los valores arrojados al final correspondiente al r cuadrada, se puede observar como el modelo después de la época 500 comienza a generar errores tanto de entrenamiento como de validación prácticamente constantes lo que sugiere que los parámetros no se están ajustando de manera significativa por lo que sería bueno tomar decisiones en si es necesario ajustar el dataset agregando información o aplicando transformaciones de la información, así mismo considerar cambiar hiperparametros. Por otro lado viendo los valores de r cuadrada indica que el modelo no genera datos confiables por tener un valor bajo, por lo tanto reafirma lo que se mencionaba anteriormente, indicando que se deben de hacer cambios para la mejora del modelo.

Finalmente se agrega la siguiente gráfica como representación de los datos que se esperan comparado con los valores predichos, aquí se puede ver que el modelo no está comprendiendo los datos correctamente arrojando incluso datos negativos lo cual no es posible que ocurra bajo este modelo de datos.



Regresión lineal con framework

En esta implementación se usó el framework de Scikit-Learn para construir y evaluar el modelo de regresión lineal previamente analizado de manera más eficiente y estructurada. A diferencia de la implementación manual esta versión aprovecha las herramientas avanzadas de Scikit-Learn para la división en conjuntos de entrenamiento, validación y prueba, así como para la evaluación del rendimiento del modelo.

Mejoría del ETL

En la revisión y mejora del procesamiento de datos para el modelo de regresión se implementaron varios cambios clave para mejorar la calidad del dataset. Inicialmente se cargó el dataset original y se simplificó, enfocándose en las columnas esenciales para el análisis: ‘Hour’, ‘Temperature’, ‘Seasons’ y ‘Rented Bike Count’ y se realizó una conversión de valores categóricos en la columna ‘Seasons’ a valores numéricos para facilitar el análisis.

Sin embargo dado que el modelo manual mostró un desempeño poco confiable se decidió realizar una actualización del ETL realizado al dataset, donde los cambios incluyeron la renumeración de las columnas para mejorar la claridad de los datos, se eliminaron columnas innecesarias, como ‘Functioning Day’ ya que se tenían datos que no ayudaban al ser prácticamente los mismos, y se des duplicaron los datos para asegurar que cada registro fuera único además la columna de fecha se procesó para extraer solo el mes, simplificando así la representación temporal pero manteniendo información que puede llegar a ser útil, además, se aplicó One-Hot Encoding para convertir la columna ‘Season’ en variables dummy representando cada estación como una columna binaria separada. Por otro lado, la columna ‘Holiday’ también se transformó en valores binarios para reflejar la presencia o ausencia de días festivos.

Random forest

Random Forest es un algoritmo de aprendizaje supervisado que combina múltiples árboles de decisión para mejorar la precisión y estabilidad del modelo, funciona creando varios subconjuntos aleatorios del conjunto de datos y construyendo un árbol de decisión en cada subconjunto para que la predicción final sea el promedio de todas las predicciones para el caso de una regresión o la mayoría de votos para clasificación, además esta técnica reduce la varianza y el riesgo de sobreajuste, haciendo que el modelo sea más robusto y preciso

Implementación

En la implementación inicial, se utilizó una Regresión Lineal para predecir el número de bicicletas alquiladas, lo que incluía cargar los datos, dividirlos en conjuntos de entrenamiento, validación y prueba, escalar los datos, entrenar el modelo y luego calcular el error cuadrático medio y el R^2 para evaluar el rendimiento del modelo, sin embargo incluso con la actualización del ETL y el uso de un framework los resultados aún parecían poco confiables, aunque si se notó una mejoría, pasando de un valor de alrededor de 39% en la implementación manual a aproximadamente 65% con la actualización del ETL y uso de framework, sin embargo se busca mejorar la calidad de las predicciones.

Para mejorar el rendimiento del modelo y hacerlo más confiable, se implementó un random forest en lugar de la regresión lineal, lo que implicó varias modificaciones y mejoras:

1. A diferencia de la regresión lineal que busca una relación lineal entre las características y la variable objetivo el random forest construye múltiples árboles de decisión lo que permite capturar relaciones no lineales más complejas entre las variables, lo que es especialmente útil para datos con mucha variabilidad o con comportamientos no lineales
2. Uso de bagging: se decidió la implementación de bagging a raiz de una detección de overfitting en el modelo y de una alta varianza, lo que significa que

entrena cada árbol en un subconjunto aleatorio de los datos, ayudando a reducir la varianza y evitando que el modelo se ajuste demasiado a datos específicos de entrenamiento o bien dicho como el overfitting, esta técnica también reduce la dependencia de un solo árbol lo que aumenta la estabilidad de las predicciones

3. En lugar de hacer una única predicción basada en un modelo como en la regresión lineal, random forest genera predicciones múltiples y las combina promediando los resultados lo que ayuda a reducir la posibilidad de errores significativos y ofrece predicciones más precisas
4. Parámetros ajustables y optimización con grid search: Para mejorar aún más el rendimiento del modelo se usó GridSearchCV para encontrar los mejores hiperparámetros del random forest, como el número de árboles (n_estimators) o la profundidad máxima de los árboles (max_depth) esto asegura que el modelo esté optimizado para el conjunto de datos en cuestión, mejorando tanto la precisión como la generalización a nuevos datos.

Grid search CV y Bagging

GridSearchCV es una técnica de búsqueda de hiperparámetros que permite encontrar la combinación óptima de estos en un modelo, lo que hace es explorar un conjunto definido de parámetros, como el número de árboles en el bosque (n_estimators), la profundidad máxima de los árboles (max_depth), o la cantidad de características evaluadas en cada división (max_features), entre otros, haciéndolo para cada combinación de hiperparámetros, entonces el modelo es entrenado y validado varias veces utilizando validación cruzada, en la cual los datos se dividen en varias partes o "folds", y el modelo se entrena y evalúa en cada uno de ellos, lo que ayuda a medir su desempeño en diferentes subconjuntos de los datos. Al usar GridSearchCV el modelo se ajusta con los mejores valores de los hiperparámetros lo que mejora la precisión y reduce el riesgo de sobreajuste (overfitting), ya que se garantiza que el modelo esté adecuadamente validado en diferentes porciones de los datos antes de elegir la mejor configuración

Por otro lado, el bagging es una técnica utilizada por random forest para reducir la varianza del modelo y mejorar su capacidad de generalización. En el bagging se crean múltiples subconjuntos aleatorios del conjunto de entrenamiento con reemplazo lo que significa que algunas observaciones se pueden repetir en diferentes subconjuntos, para luego entrenar un modelo en cada subconjunto y se promedian los resultados para hacer una predicción final, al combinar los resultados de múltiples árboles entrenados en diferentes subconjuntos de los datos, bagging hace que el modelo sea más preciso frente a pequeñas variaciones en los datos de entrenamiento evitando que se ajuste demasiado a ellos (overfiting)

Resultados obtenidos

En los gráficos resultantes del modelo de random forest, se puede analizar varios aspectos clave del rendimiento del modelo. En primer lugar el modelo muestra un bajo grado de sesgo (bias) lo que indica que es capaz de captar las relaciones complejas entre las características y la variable objetivo sin simplificar demasiado el patrón de los datos lo cual es una señal positiva ya que un sesgo bajo implica que el modelo tiene una buena capacidad para representar adecuadamente los datos, además el modelo indica una varianza media, lo que sugiere que si bien el modelo se ajusta bien a los datos de entrenamiento, existe un grado moderado de varianza en los resultados cuando se expone a datos nuevos o no vistos, donde una varianza media puede indicar que el modelo tiene cierta sensibilidad a los datos de entrenamiento, lo cual podría derivar en una ligera tendencia hacia el sobreajuste (overfitting) aunque en este caso se puede decir que se mitigó hasta cierto punto con las estrategias mencionadas anteriormente que ayudarían a reducir la varianza y overtifing, ya que en los random forest para hacer esto, equivalente a una técnica de regularización es modificar los hiperparametros, generalmente reduciendo la profundidad de los árboles y agregando nuevos, implicando que no exista tanta especialización.

Comparación de resultados

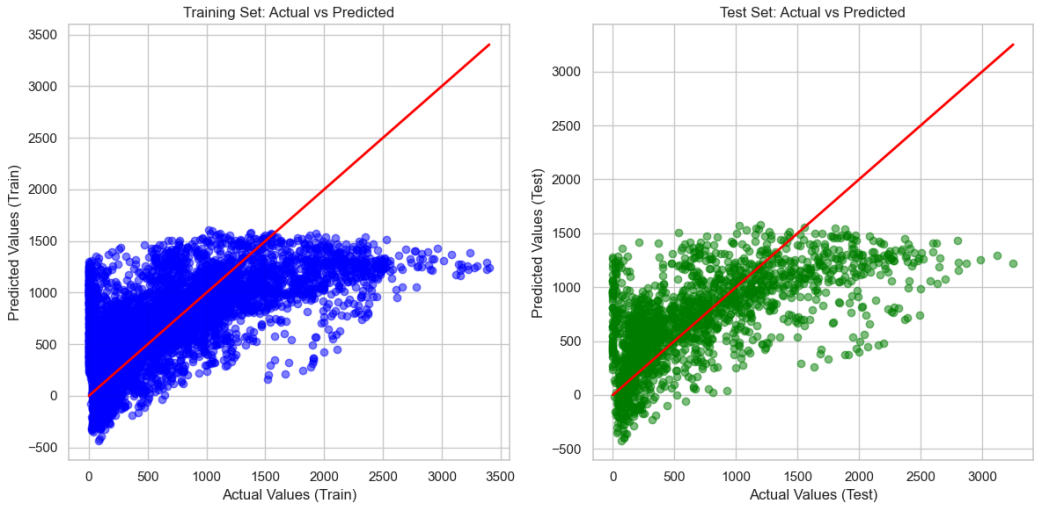
En mi implementación inicial del modelo sin utilizar ningún framework, construí un modelo de regresión básico manualmente. Utilicé técnicas estadísticas sencillas y creé un flujo de procesamiento de datos personalizado, seguido de un entrenamiento de un modelo de regresión lineal simple. Aunque este enfoque sirvió como base para entender el conjunto de datos y las relaciones entre las variables, los resultados fueron limitados. El modelo tenía dificultades para capturar patrones no lineales, y las métricas de rendimiento como el R^2 y el Error Cuadrático Medio (MSE) indicaban que el modelo no estaba generalizando bien en datos no vistos. Además, noté claros signos de overfitting, donde el modelo funcionaba bien con los datos de entrenamiento, pero mucho peor con los datos de validación y prueba. Esto mostraba que el modelo estaba demasiado ajustado a los datos de entrenamiento y no tenía la flexibilidad necesaria para generalizar.

Después, se refactorizo el modelo utilizando el framework scikit-learn, lo que mejoró el flujo de trabajo al automatizar muchos procesos como el escalado, la validación cruzada y la evaluación. Aunque el modelo de regresión lineal construido dentro del framework mostró ligeras mejoras en las métricas de rendimiento y facilitó la afinación de hiperparámetros a través de GridSearchCV, el problema del overfitting persistía. El modelo seguía teniendo dificultades con la complejidad de los datos, y a pesar de las optimizaciones, la mejora fue modesta. El framework hizo el proceso más eficiente, pero los problemas fundamentales del modelo permanecían.

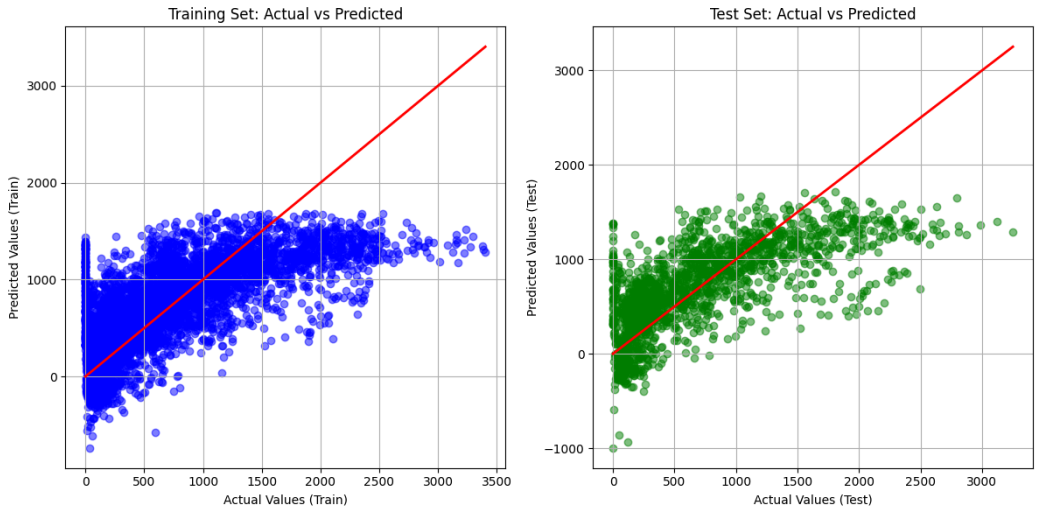
Reconociendo las limitaciones del modelo lineal, especialmente el overfitting, cambié el enfoque hacia una técnica de aprendizaje automático más avanzada: Random Forest. Este método de ensamble mejoró drásticamente los resultados. Random Forest fue capaz de capturar patrones complejos y no lineales que los modelos lineales no habían detectado, y su regularización inherente ayudó a mitigar el overfitting. Las métricas de rendimiento mostraron una mejora significativa, con un mayor R^2 y una notable reducción en el MSE. La búsqueda de hiper parámetros a través de GridSearch también optimizó el modelo, resultando en una solución robusta y bien generalizada. Además, la incorporación de técnicas como Bagging añadió aún más estabilidad,

haciendo que este enfoque fuera mucho más superior en comparación con las implementaciones anteriores.

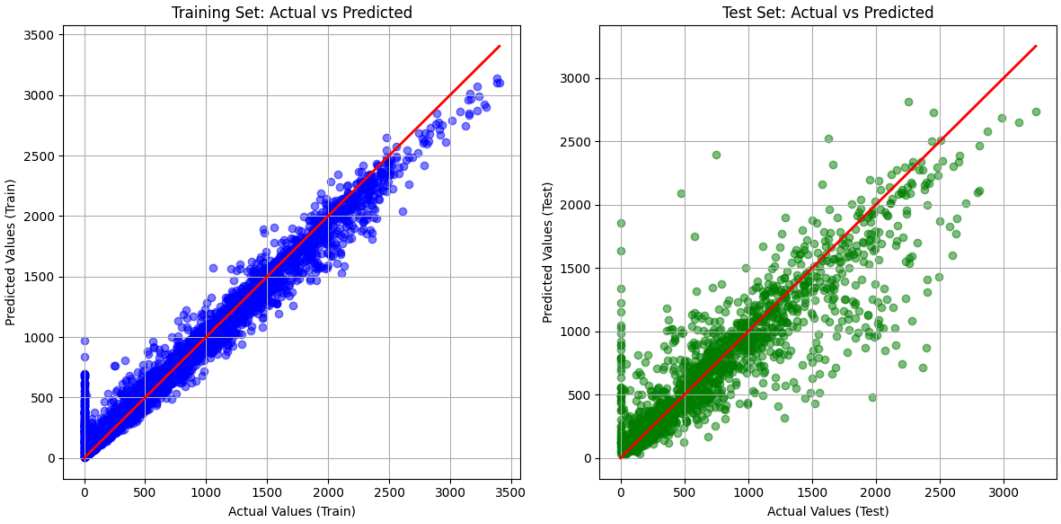
Comparación de resultados predichos contra esperados con implementación a mano



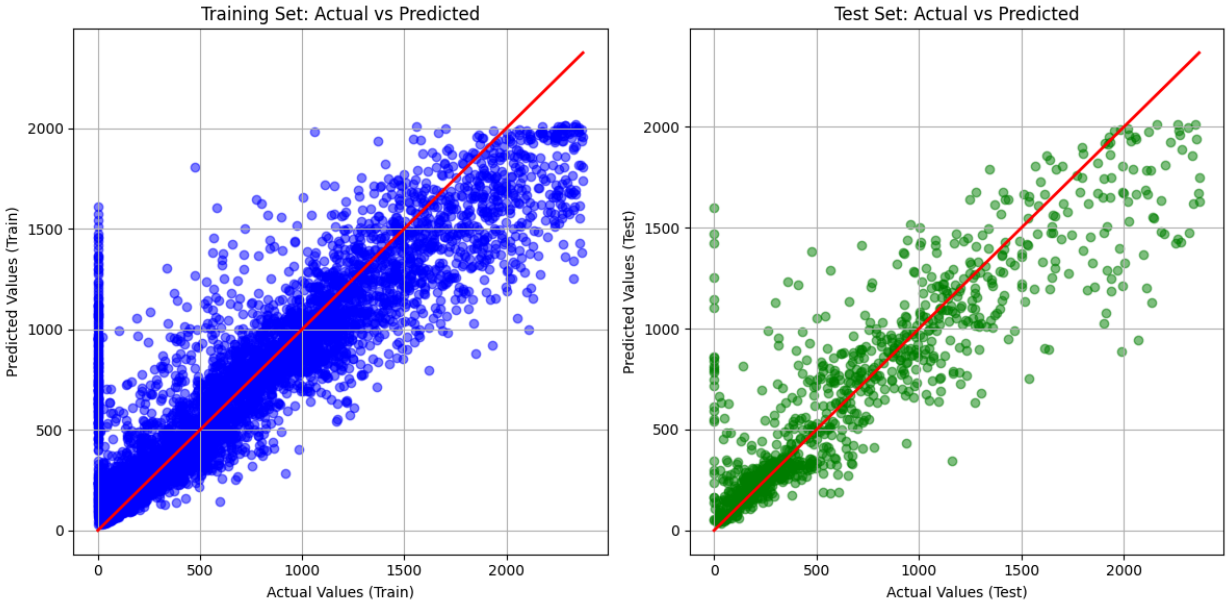
Comparación de resultados predichos contra esperados con uso de framework



Comparación de resultados predichos contra esperados con uso de random forest con overfitting y una alta variabilidad



Comparación de resultados predichos contra esperados con uso de random forest con overfitting solucionado parcialmente con bagging y probar con diferentes hiperparametros



Referencias

¿Qué es la regresión lineal? - Explicación del modelo de regresión lineal - AWS. (n.d.). Amazon Web Services, Inc. https://aws.amazon.com/es/what-is/linear-regression/

Libretexts. (2024, March 17). *6: Optimización y Método del Gradiente Descendiente*. Mathematics LibreTexts. https://math.libretexts.org/Courses/Universidad_Complutense_de_Madrid/Las_matematicas_de_la_inteligencia_artificial/06%3A_Optimizacion_y_Metodo_del_Gradiente_Descendiente#:~:text=solucionar%20este%20problema.-,M%C3%A9todo%20del%20gradiente%20descendente%20estoc%C3%A1stico,la%20direcci%C3%B3n%20de%20m%C3%A1ximo%20descenso

LinearRegression. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

IBM Cognos Analytics 11.1.x. (n.d.). https://www.ibm.com/docs/es/cognos-analytics/11.1.0?topic=terms-r2

Stewart, K. (2023, March 22). *Mean squared error (MSE) | Definition, Formula, Interpretation, & Facts*. Encyclopedia Britannica. https://www.britannica.com/science/mean-squared-error

What is Random Forest? | IBM. (n.d.). https://www.ibm.com/topics/random-forest#:~:text=Random%20forest%20is%20a%20commonly,both%20classification%20and%20regression%20problems

GridSearchCV. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

Navarro, S. (2024, April 16). ¿Qué es GridSearchCV? | KeepCoding Bootcamps. *KeepCoding Bootcamps*. https://keepcoding.io/blog/que-es-gridsearchcv/

BaggingRegressor. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingRegressor.html

¿Qué es una red neuronal? | *IBM*. (n.d.). https://www.ibm.com/mx-es/topics/neural-networks