“Inteligencia Artificial en la cadena de suministros para deliveries motorizados en la ciudad de Lima”

Autor/Autores: Aarón U. Santillán H., Jim Navarrete C., Ludwin N. Cahuaya C., Breidy E. Estrada M.

Resumen-El siguiente trabajo tiene como objetivo evaluar un modelo de Machine Learning y Redes Neuronales para la obtención de la mejor ruta de un delivery para la entrega de suministros en la ciudad de Lima. Para este objetivo se proponen distintos algoritmos de regresión en el caso de machine learning y redes multicapa en el caso de redes neuronales. Se procede a realizar el entrenamiento con estos algoritmos para seleccionar el mejor de cada en función de ciertas métricas de rendimiento y que se pueden visualizar mejor mediante boxplots. Luego se evalúa la data ‘test’ con el mejor modelo. Finalmente, el modelo de machine learning obtuvo mejor rendimiento respecto de las redes neuronales con un R2score de 0.81.

1. Introducción

- **Descripción del problema: Contexto, relevancia y justificativa**

En los últimos años, el comercio electrónico ha experimentado un crecimiento significativo, adquiriendo mayor relevancia a raíz de la pandemia de COVID-19. La tendencia de los consumidores a realizar compras en línea y solicitar servicios de entrega a domicilio ha incrementado sustancialmente. Según el diario ‘El Comercio’ este creció 55% en el 2021 al mover US$ 9.3 millones. Como consecuencia, las empresas han tenido que adaptarse para gestionar un mayor volumen de pedidos y garantizar un servicio eficiente de entrega a domicilio. En muchos casos, la propia empresa asume la responsabilidad de este servicio, lo cual implica costos adicionales considerables. Estos negocios suelen enviar sus pedidos mediante empresas de delivery como Rappi, que suelen cobrar 15% - 20% del total del envío. Estos costos están directamente relacionados con la cantidad de pedidos, el consumo de combustible necesario para realizar las entregas, y la planificación de las rutas a seguir.

- **Hipótesis y/o pregunta a abordar**

Si una empresa contara con información precisa sobre el orden óptimo para realizar las entregas, considerando factores como el tráfico, las distancias entre los puntos de entrega y el tiempo total requerido para completar la ruta, ¿sería posible reducir los costos asociados al servicio de delivery haciendo uso de inteligencia artificial para lograr predecir el mejor orden de entrega?

- **Objetivos**

Desarrollar un algoritmo de inteligencia artificial que determine el orden de entrega de los pedidos, con el fin de minimizar los costos operativos asociados al servicio de delivery.

2. Trabajos relacionados

- **A machine learning optimization approach for last-mile delivery and third-party logistics**

Se aborda el problema VCSBPPSI (Variable Cost and Size Bin Packing Problem with Stochastic Items), que implica decisiones logísticas en dos etapas. En la etapa táctica, se decide la capacidad de transporte con un proveedor externo (3PL) antes de conocer la demanda exacta, minimizando costos a futuro. En la etapa operativa, se ajusta la capacidad, incurriendo en costos adicionales si la demanda real excede las reservas previas.

Este enfoque introduce una heurística basada en ML para clasificar cada contenedor según su probabilidad de ser óptimo en la solución final. A través de un clasificador supervisado, se recopilan características (costo relativo, capacidad, entre otras) que el modelo utiliza para determinar qué contenedores reservar. Este clasificador se entrena con datos históricos de decisiones óptimas, permitiendo ajustar los recursos en instancias futuras de manera precisa.

Utilizando valores SHAP, el modelo ML identifica las características más influyentes para su predicción. Estas incluyen el valor de la relajación continua y el costo reducido, que ayudan a evaluar qué contenedores se deben reservar anticipadamente. Esta selección optimiza el rendimiento del clasificador al enfocarse en las variables con mayor impacto en la decisión.

El PH, un algoritmo de optimización tradicional que utiliza relajación de Lagrange, se utiliza como referencia. Este algoritmo es efectivo en problemas complejos, aunque requiere mayor tiempo de procesamiento que la heurística de ML. Se comparan ambos métodos en diversos escenarios, demostrando que el modelo ML es más rápido,

adecuado para ajustes en tiempo real, mientras que el PH ofrece mayor precisión en entornos logísticos de alta complejidad.

- **A study on the pickup and delivery problem with time windows: Matheuristics and new instances**

Este artículo presenta un enfoque innovador que integra técnicas de Machine Learning (ML) para abordar el "Pickup and Delivery Problem" (PDP) con ventanas de tiempo. La metodología del estudio comienza con la definición del problema, considerando tanto los puntos de recogida como de entrega, e incorporando restricciones temporales. Para ello, los autores recopilan datos históricos de operaciones previas que incluyen tiempos de entrega, distancias recorridas y condiciones de tráfico, lo que permite entrenar modelos de regresión, como regresión lineal y árboles de decisión, para predecir los tiempos de viaje en función de diversas características del trayecto.

Una vez entrenado, el modelo se utiliza para estimar los tiempos de viaje y los costos asociados a nuevas rutas. Además, se aplican algoritmos de optimización, como la programación entera, que se benefician de las predicciones generadas por el modelo de ML, lo que resulta en una planificación más eficiente de las rutas. Los resultados del estudio destacan la alta precisión del modelo en las predicciones, lo que se traduce en una significativa reducción de los costos operativos y mejoras en la puntualidad de las entregas. En comparación con métodos tradicionales de optimización, la integración de ML se muestra ventajosa, permitiendo soluciones más rápidas y precisas.

Los autores concluyen que la combinación de técnicas de ML y algoritmos de optimización no solo mejora la eficiencia en el PDP, sino que también permite una mayor adaptabilidad a cambios dinámicos en las condiciones de entrega, lo que es fundamental para las operaciones logísticas en tiempo real.

- **Predicting Drivers' Route Trajectories in Last-Mile Delivery Using a Pair-wise Attention-based Pointer Neural Network**

El artículo se centra en cómo las redes neuronales pueden optimizar las rutas de entrega en la última milla, un aspecto crucial para la eficiencia logística. En este estudio, se presenta un modelo basado en una Pointer Neural Network (PNN), combinada con un mecanismo de atención pareada. Este enfoque permite predecir las trayectorias reales de los conductores, considerando sus desviaciones sistemáticas respecto a las rutas óptimas planeadas. La atención pareada facilita la evaluación de la relación entre cada par de nodos (puntos de entrega), priorizando decisiones de enrutamiento más precisas.

El modelo se entrena utilizando datos históricos y en tiempo real, lo que le permite adaptarse dinámicamente a las condiciones cambiantes del tráfico y las decisiones de los conductores. Esto mejora la eficiencia en la asignación de rutas y reduce tanto los costos operativos como los tiempos de entrega. Además, el modelo se valida utilizando conjuntos de datos reales del sector logístico, demostrando que supera los enfoques tradicionales de optimización de rutas.

Este enfoque se destaca por su capacidad para capturar patrones complejos en las rutas de entrega, lo que lo hace aplicable a sistemas logísticos de última milla en entornos urbanos, donde las condiciones son más dinámicas y difíciles de predecir.

3. Metodología

3.1 Descripción del problema

El artículo busca solucionar el problema de encontrar una ruta óptima para la entrega de deliveries en la ciudad de Lima.
Para este objetivo se probarán distintos algoritmos de machine learning y redes neuronales, los cuales son usualmente usados para problemas de búsqueda con la mejor opción para encontrar resultados óptimos.

3.2. Entrada y Salida

Se utilizará data proporcionada por Shutar. S [4], en su artículo en Kagle se tiene los siguientes atributos:

#	Column	Non-Null Count	Dtype
0	Order_ID	43739 non-null	object
1	Agent_Age	43739 non-null	int64
2	Agent_Rating	43685 non-null	float64
3	Store_Latitude	43739 non-null	float64
4	Store_Longitude	43739 non-null	float64
5	Drop_Latitude	43739 non-null	float64
6	Drop_Longitude	43739 non-null	float64
7	Order_Date	43739 non-null	object
8	Order_Time	43739 non-null	object
9	Pickup_Time	43739 non-null	object
10	Weather	43648 non-null	object
11	Traffic	43739 non-null	object
12	Vehicle	43739 non-null	object
13	Area	43739 non-null	object
14	Delivery_Time	43739 non-null	int64
15	Category	43739 non-null	object

Figura 1. *Atributos y Tipos de datos del csv inicial.*

No obstante, para realizar el entrenamiento del modelo la data debe de pasar por una etapa de

preprocesamiento, los cuales pasan por las siguientes etapas:
Primero se identifica si los atributos tienen valores nulos o si se debe imputar algunos valores.
Luego se identifica si todos los atributos son del tipo de dato que deseamos para que se pueda entrenar. En la tabla 1 se muestran los tipos de datos que deberían tener cada atributo.

Tabla 1. *Tipo de dato de atributos*

Atributo	Tipo de dato
Agent_Age	float64
Agent_Rating	float64
Store_Latitude	float64
Store_Longitude	float64
Drop_Latitude	float64
Drop_Longitude	float64
Weather	float64
Traffic	float64
Vehicle	float64
Area	float64
Delivery_Time	float64
Category	float64

Mediante label_encoding se aplicará a Weather, Traffic, Vehicle, Category y Área, de modo que sus valores cambian a números para usarlo en el entrenamiento del modelo.
En el caso de los atributos de coordenadas 'Store' y 'Drop', estos se sustituyen por otro atributo 'Distance' mediante un método para convertir a distancia Haversine.
Esta metodología de preprocesamiento se realizó similar a la metodología de preprocesamiento para predecir el delivery time de Amazon usando regresión (Suthar, 2024).

Por último, mediante una matriz de correlación de esta data se puede apreciar lo siguiente:

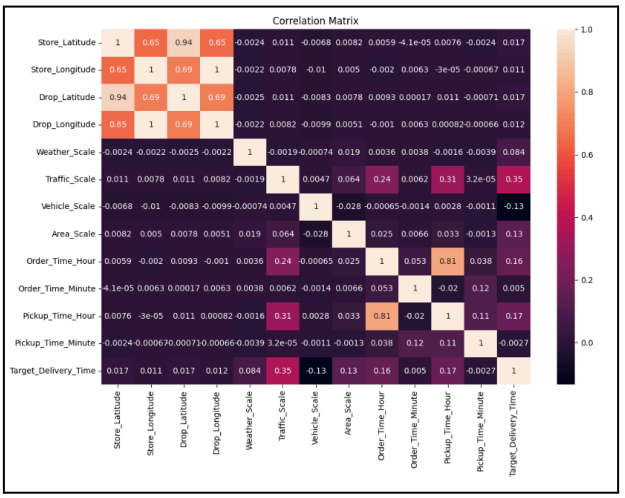

Figura 2. *Matriz de Correlación entre la data preprocesada.*

Posteriormente se establece como target a la variable "Delivery_Time" y se asignan a las demás columnas como atributos. Se realiza una partición del conjunto de datos en conjuntos de training y test (80% y 20%).
Para el caso de machine learning se propusieron varios algoritmos de regresión para evaluar el target. En este trabajo se emplearon los siguientes:

- Regresión Lineal: Este modelo asume una relación lineal entre las variables y busca minimizar el error cuadrático ajustando los coeficientes.
- Regresión Ridge. Es una extensión del método de regresión lineal pero añade un término de regularización L2 . Esta regularización penaliza los coeficientes grandes. Esto ayuda a controlar el sobre ajuste y estabilidad del modelo en multicolinealidad.
- Lasso: Es similar al anterior modelo, pero este utiliza un término de regularización L1. Este modelo tiene la capacidad de llevar a algunos coeficientes a cero, seleccionando las variables más importantes y simplificando el modelo.
- ElasticNet: Este modelo combina los métodos Ridge y Lasso al utilizar tanto la regularización L1 y L2. ElasticNet busca un equilibrio entre los dos métodos, obteniendo un modelo robusto que selecciona los features más significativos y balancea los coeficientes.
- K-Neighbors Regressor (KNN): Es un modelo que predice el valor del target basándose en el promedio de los k-vecinos más cercanos en el espacio de features. Es útil para patrones no lineales.
- Decision Trees: Este modelo se basa en dividir los datos en subconjuntos en función de características específicas. Se trata de que los árboles no sean tan

profundos para no sobre ajustar el modelo a la data de entrenamiento. Son útiles en relaciones no lineales.
- Random Forest: Se encarga de combinar múltiples árboles de decisión entrenados en diferentes subconjuntos.
- Light Gradient Boosting Machine: Este modelo crea árboles de decisión basados en hojas para optimizar la velocidad y la precisión de grandes conjuntos de datos.
- Extreme Gradient Boosting: Mediante este algoritmo se realiza un aumento de gradiente, ajustando el modelo iterativamente para reducir el error de predicción residual.
- Extremely Randomized Tree: Crea árboles de decisión extremadamente aleatorios para mejorar la generalización al momento de la mayor variación de splits.

Luego se aplicó una estrategia de cross-validation de 10 k-folds. Esta técnica permite evaluar los modelos de manera más robusta, ya que se entrenan y prueban en diferentes subconjuntos del conjunto de entrenamiento. La métrica utilizada para la evaluación del error fue el error cuadrático medio negativo. Luego se escogerá el modelo cuya mediana de error sea menor y además tenga menor varianza, obteniendo precisión y estabilidad.

Por otro lado, para el caso de redes neuronales, se propondrán distintas capas para luego evaluarlas bajo la misma métrica con la que se evaluó en machine learning y se obtendrá la mejor red multicapa, para luego reentrenar el mejor modelo con los data test.

Finalmente, se generará un gráfico de dispersión que muestre la relación entre los valores reales y los valores predichos del modelo. En este gráfico, se determinará con qué precisión predice las técnica propuestas.

4. Experimentación y Resultados

Setup experimental:

- Datos empleados:

Se utilizan los siguientes atributos para realizar el experimento, pues de acuerdo a la matriz de correlación presentada anteriormente, son los atributos que aportan información relevante:

Tabla 2. *Atributos*

Atributo
Agent_Age
Agent_Rating
Distance
Weather_encoded
Area_encoded
Traffic_encoded
Vehicle_encoded
Category_encoded

- Métricas de evaluación:

Al ser un problema de regresión se escoge como métrica de evaluación 'Mean Absolute Error' para redes neuronales y 'neg_mean_squared_error' para machine learning ya que penaliza severamente los grandes errores, lo que ayuda a optimizar modelos de predicción al ajustar sus parámetros de manera eficiente. Esta métrica es útil en problemas de regresión, ya que minimiza la discrepancia entre los valores predichos y reales. Al ser diferenciable, facilita la optimización mediante algoritmos como el descenso por gradiente.

- Experimentos

Algoritmos de Regresión

Se evalúa distintos algoritmos de machine learning bajo el parámetro como 'LinearRegression', 'Ridge'. 'Lasso', 'ElasticNet', 'KNN', 'DecisionTreeRegressor', 'RandomForestRegressor, 'LGBMRegressor', 'XGBRegressor', 'ExtraTreesRegressor' y 'xtree'.
Se utiliza xtree el cual es una variación de XGBRegresor con los siguientes parámetros:

```
xtree = XGBRegressor(
    n_estimators=300,           # Más árboles
    learning_rate=0.01,         # Tasa de aprendizaje más baja
     max_depth=12,                  # Profundidad de los árboles
     subsample=0.8,                 # Fracción de datos para cada árbol
    random_state=42
)
```

De los cuales se obtuvieron los siguientes resultados:

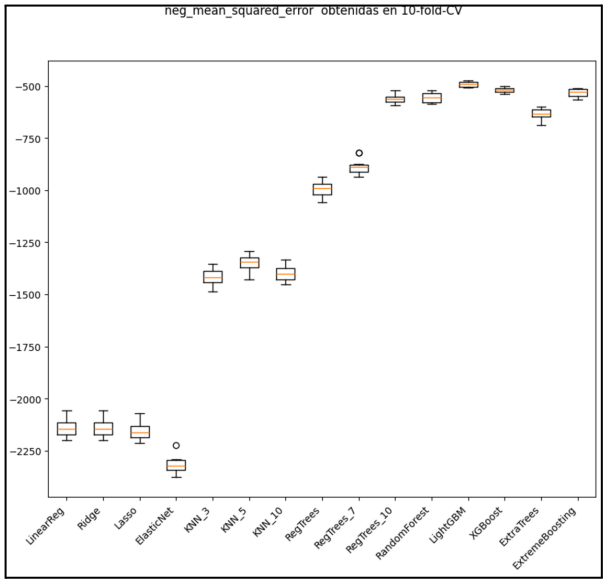

Figura 3. *Boxplots de evaluación de algoritmos*

Tabla 3. *Evaluación de algoritmos*

LinearReg	*-2139.5960*
Ridge	*-2139.5959*
Lasso	*-2155.7174*
ElasticNet	*-2317.3597*
KNN_3	*-1417.4803*
KNN_5	*-1349.3498*
KNN_10	*-1400.2041*
RegTrees	*-994.5799*
RegTrees_7	*-885.3995*
RegTrees_10	*-562.1561*
RandomForest	*-556.6310*
LightGBM	*-491.9587*
XGBoost	*-520.3287*
ExtraTree	*-633.2028*
ExtremeBoosting	*-533.4510*

Asimismo, se empleó pipelines para escalar la data y reducir dimensión, siendo estos:

Standard Scaler, MinMax Scaler, Normalizer.

PCA (5 componentes) - Se realizará en un pipeline luego de usar los escaladores correspondientes.

De este modo se obtiene los siguientes resultados:

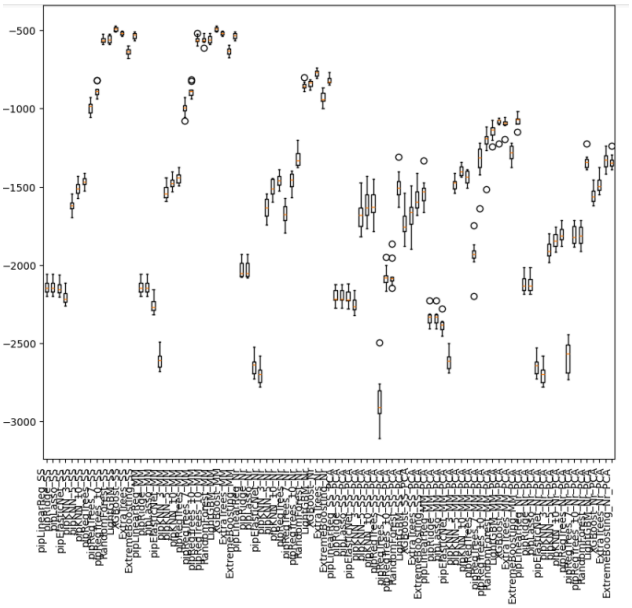

Figura 4. *Boxplots de evaluación de algoritmos con escaladores*

Tomando en cuenta estos resultados, se obtiene que el pipeline generado por Standar Scaler y LGBMRegresor es el mejor con MSE = `−491.8723781918278` y con desviación estandar = `12.451484257686083` . De modo que se entrena el modelo con 'pipeline Standar Scaler - LGBMRegressor' con la data de training.

Redes Neuronales

Se proponen distintas redes multicapa:

Tabla 4. *Redes multicapa*

Red	***Cant. de parámetros***
MLP	*4,801*
MLP1	*11521*
MLP2	*6,849*
MLP3	*6,849*
MLP4	*25,985*

Luego evaluando el 'Mean Absolute Error' se obtuvo que MLP4 es el mejor modelo.

■ Resultados y Discusión:

Algoritmos de Regresión

Se obtuvieron las siguientes métricas de rendimiento cuando se validó con la data de test.

Tabla 5. *Métricas de machine learning*

Métrica	**Resultado**

Mean squared error	501.4211
Mean absolute error	17.5619
Explained variance score	0.8119
R2 score	0.8119

A continuación se muestra una representación gráfica de los valores reales vs los valores predichos. Se ve que los valores están cerca de la línea, por lo que el modelo está funcionando correctamente.

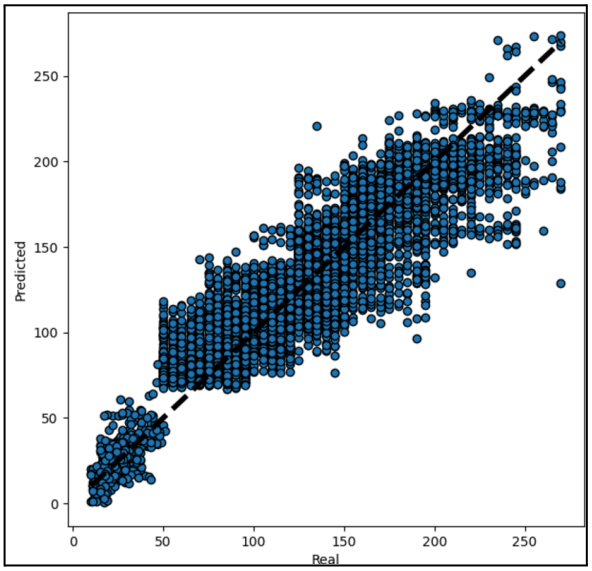

Figura 5. *Boxplots de evaluación de algoritmos*

Luego, para intentar lograr mejorar las métricas se utilizó distintas técnicas de escalamiento y pipelines, obteniendo:

Tabla 6. *Métricas empleando pipelines*

Métrica	**Resultado**
Mean squared error	499.3882
Mean absolute error	17.5357
Explained variance score	0.8127
R2 score	0.8127

Redes Neuronales

Se obtuvieron las siguientes métricas de rendimiento cuando se validó con la data de test.

Tabla 7. *Métricas de redes neuronales MLP4*

Métrica	**Resultado**
Mean squared error	540.3178
Mean absolute error	17.9114
Explained variance score	0.7951
R2 score	0.7945

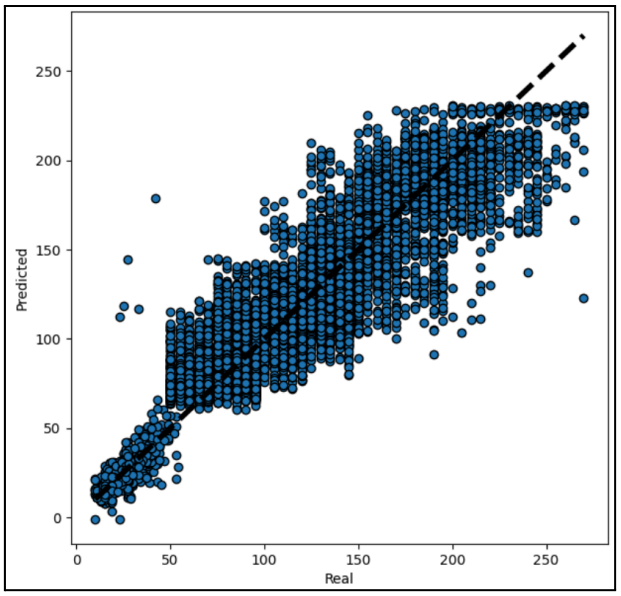

Figura 6. *Boxplots de evaluación de red MLP4*

5. Conclusiones

- Utilizar el atributo 'Distance' en lugar de coordenadas "Store' y 'Drop' produjo mejor rendimiento al momento de entrenar el modelo.
- Evaluando ambas técnicas, machine learning con algoritmo de regresión con pipelines demostró tener mejor rendimiento respecto del uso de redes neuronales según la métrica R2 score.
- La implementación de un algoritmo A* en combinación con un modelo de predicción de tiempos muestra potencial para reducir costos operativos y mejorar la eficiencia del servicio. Ya que permite predecir el tiempo de entrega estimado y de esta manera planificar mejor el tiempo de entrega.
- La prueba de diferentes estrategias de regresión y cross validation ayudaron a identificar el mejor modelo de predicción

6. Sugerencias de trabajos futuros

- Entrenar el modelo con datos que tengan un atributo de distancia real en lugar de estimarla.
- Entrenar el modelo con data de la ciudad en la cual será desplegada, en este caso Lima.
- Para el caso de redes neuronales, explorar más arquitecturas y formas de construir los modelos
- Considerar en el entrenamiento los puntos específicos por donde el agente de delivery debe pasar

- Considerar las calles y direcciones válidas de navegación del agente de delivery

7. Implicancias éticas

- **Sesgos en los datos**
  Los modelos entrenados con datos históricos pueden heredar sesgos presentes en estos, como preferencias involuntarias hacia ciertas áreas o tipos de pedidos. Esto podría resultar en discriminación hacia zonas desfavorecidas o condiciones específicas de entrega. Para mitigar esto, es fundamental realizar auditorías de equidad en los datos y los modelos de forma periódica.

- **Privacidad y seguridad de los datos**
  La recopilación y procesamiento de datos sensibles, como ubicaciones y horarios de usuarios, requieren estrictas políticas de seguridad para evitar violaciones de privacidad o uso indebido de la información. Se deben implementar medidas como la anonimización de datos.

- **Transparencia y explicabilidad**
  Los usuarios finales, tanto repartidores como empresas, deben tener acceso a explicaciones claras sobre cómo el sistema toma decisiones. Esto refuerza la confianza en el modelo y permite corregir errores o malas prácticas.

8. Link del repositorio del trabajo

https://drive.google.com/file/d/1rHH8BiFVp-DvDJa2SE4asuSyiksR_p4F/view?usp=sharing

Presentación: Link de diapositivas

9. Declaración de contribución de cada integrante

Jim: Entrenamiento con redes neuronales
Aaron: Entrenamiento con algoritmos de regresión
Edmara: Redacción de algoritmos en función de los algoritmos entrenados

Nereo: Revisión general del documento y redacción de conclusiones y sugerencias

10. Referencias

[1]. Bruni, ME, Fadda, E., Fedorov, S. y Perboli, G. (2023). Un enfoque de optimización de aprendizaje automático para la entrega de última milla y la logística de terceros. *Computers & Operations Research, 157* , 106262. https ://doi.o/1/j.cor.2023.106262

[2]. Sartori, CS y Buriol, LS (2020). Un estudio sobre el problema de recogida y entrega con ventanas de tiempo: matemáticas y nuevas instancias. Computers & Operations Research, *124* , 105065. https://doi.org/10.1016/j.cor.2020.105065

[3]. Amazon Delivery Dataset. (2024, 2 julio). Kaggle. https://www.kaggle.com/datasets/sujalsuthar/amazon-delivery-dataset

[4]. Directions API overview. (s. f.). Google For Developers. https://developers.google.com/maps/documentation/directions/overview

[5]. El Comercio. (2022, 28 de enero). *Comercio electrónico creció 55% en el 2021 al mover US$ 9300 millones, según CAPECE*. Recuperado de https://elcomercio.pe/economia/peru/comercio-electronico-crecio-55-en-el-2021-al-mover-us-9300-millones-segun-capece-rmmn-noticia/

[6]. Mo, B., Wang, Q., Guo, X., Winkenbach, M., & Zhao, J. (2023). Predicting drivers' route trajectories in last-mile delivery using a pair-wise attention-based pointer neural network. *Transportation Research Part E: Logistics and Transportation Review*, 175. https://doi.org/10.1016/j.tre.2023.102084

[7]. Suthar, S. (2024, 12 de Julio). *Predicting Amazon delivery time using regression*. Kaggle. Recuperado de https://www.kaggle.com/code/sujalsuthar/predicting-amazon-delivery-time-using-regression