



## Propuesta de modelos ML para la empresa TaxiCom 2.0

Título	Objetivo	Input. Datos de entrada consumidos en BigQuery.	ML	OutPut. Entregable
<b>Análisis de demanda por hora</b>	Predecir las zonas y horarios con mayor demanda.	(Zonastaxi, taxis_amarillos, taxis_verdes)	Modelos de regresión: Random Forest, XGBoost o LightGBM. Clustering (K-Means, DBSCAN): Para segmentar zonas de alta/baja demanda. Series temporales: ARIMA, SARIMA, o Prophet, para predecir demanda horaria.	Mapa de calor con zonas y horarios de mayor demanda. Predicciones de demanda diaria/semanal para mejorar asignación de recursos. Predicción de consumo de combustible por tipo de vehículo y ruta. Identificación de vehículos menos eficientes y recomendaciones de mejora Reportes de impacto ambiental de taxis y vehículos por tipo de combustible. Mapas de zonas con mayor contaminación generada. Mapa de calor con zonas y horarios de mayor demanda. Predicciones de demanda diaria/semanal para mejorar asignación de recursos. Identificación de vehículos más eficientes y menos contaminantes. Reportes de recomendaciones para reemplazo de vehículos.
<b>Predicción de profit por zonas</b>	Establecer una predicción de ganancia para taxis amarillos por zona.	Taxis Amarillos		<ul style="list-style-type: none"><li>- Mejores zonas de ganancias</li><li>- Predicción en ganancia por cantidad de viajes.</li></ul>
<b>Contaminación de los taxis</b>	Correlación entre demanda de taxis por zonas y contaminación del aire producida por vehículos.			<ul style="list-style-type: none"><li>- Gráficos de regresión</li><li>-</li></ul>
<b>Sistema de recomendación de zonas</b>	Recomendar zonas de acuerdo a los días de la semana, el mejor horario, ganancia en taxis	Yellow_Tripdata_2024-10.csv transformed_taxi_zone_merged_with_locations.csv	<b>XGBoost (XGBRegressor)</b> : Hiperparámetros utilizados: n_estimators=200: Número de árboles a construir. max_depth=6: Profundidad máxima de cada árbol. learning_rate=0.1: Tasa de aprendizaje que controla cuánto cambia el modelo en cada	<ul style="list-style-type: none"><li>- La api ofrece de acuerdo al día seleccionado la siguiente información<ul style="list-style-type: none"><li>- La zonas con mayor demanda</li><li>- La hora de dicha demanda</li><li>- Ganancia promedio según la zona, el hora de día y día de la semana</li><li>- La distancia promedio que se recorrerá</li><li>- Cantidad de viajes registrados en la zona</li></ul></li></ul>



	amarillos y para taxis verdes.		iteración. <b>train_test_split (de sklearn):</b> Divide los datos en conjuntos de entrenamiento y prueba para evaluar el rendimiento del modelo. <b>R^2 Score (coeficiente de determinación):</b> Métrica para evaluar la calidad de las predicciones del modelo. Un valor cercano a 1 indica un buen ajuste	<ul style="list-style-type: none"><li>- Un url de maps para ir a la zona.</li><li>- Un gráfico por día con la demanda por zona</li></ul>
<b>Clasificación de vehículos por eficiencia</b>	Clasificar los vehículos en cuatro categorías de eficiencia basadas en su consumo combinado	transformed_Vehicle Fuel Economy Data.csv	<ul style="list-style-type: none"><li>- Modelo utilizado: <b>Regresión lineal.</b></li></ul>	<ul style="list-style-type: none"><li>- La API clasifica los vehículos en cuatro categorías de eficiencia basadas en su consumo combinado</li><li>- Un archivo CSV que lista todos los vehículos clasificados por categoría y ordenados de mayor a menor eficiencia.</li><li>- Gráficos visuales para cada métrica clave (e.g., rango, consumo en ciudad) que permiten comparar rápidamente los vehículos más destacados.</li><li>- Predicción del costo basado en los valores promedio del modelo</li></ul>
<b>Calidad de aire en NY</b>	Análisis Histórico y predicción de la contaminación de aire (Pm 2.5)	sensores 22-24.csv	Primer modelo series de tiempo: Uso del algoritmo <b>Arima</b> Segundo modelo series de tiempo: Uso del algoritmo <b>Prophet</b>	<ul style="list-style-type: none"><li>- Graficos y analisis</li><li>- Predicción a 1 - 3 meses.</li></ul>
	Clustering Basado en Sitios según el índice de contaminación.	MergeValueLocation.scv = location.csv + sensores 22-24	Clustering: Modelos utilizados: <b>Kmeans, regresión lineal.</b>	<ul style="list-style-type: none"><li>- Api de agrupación por Índice de contaminación</li><li>- Función que devuelve, con los datos de los sensores y sitios, un mapa con la zona seleccionada en color (según el índice de métricas) y sus datos de contaminación por número de mes, día y hora.</li></ul>
	Agrupar las Boroughs en función de sus características clave	Air_quality.csv Boroughs_clusters	<ul style="list-style-type: none"><li>- Obtención del número óptimo de clusters con <b>elbow</b></li><li>- Aplicación del algoritmo <b>k means</b></li><li>- Análisis de correlación con <b>regresión lineal.</b></li></ul>	<ul style="list-style-type: none"><li>- Gráficos aleatorios de zonas con variables</li><li>- Gráfico de regresión lineal que muestra la correlación entre tránsito y pm 2.5</li><li>- Gráfico de las 10 zonas con más correlación respecto a la recta de regresión.</li></ul>



Objetivo	ML	Entregable
Conocer las zonas de alta demanda para optimización del negocio.	Regresión lineal. Kmeans	API de recomendación de zonas en base a la demanda de usuarios
Mostrar el desarrollo temporal de la calidad del aire en función de zonas.	Prophet Arima Clustering: Kmeans, regresión lineal.	Análisis Histórico y predicción de la contaminación de aire (Pm 2.5)
Prever y predecir posibles ganancias en función a zonas de demanda de taxis amarillos y verdes.	Prophet Clustering: Kmeans, regresión lineal.	Gráfico de proyección de ganancias. Función que recomienda las zonas de posibles sobreganancias.
Clasificar vehículos en base a su eficiencia y previsión de gasto.	Regresión lineal. DBSCAN	API de clasificación de vehículos en base a categorías de eficiencia.