Avance 2 proyecto

Elaborado por: Andres Santiago Serna Tangarife, Katherine Russi Parra, William Morales y Daniela Castillo Téllez

1. Resumen sobre el problema que están abordando (máx. 1 página):

1.1. Contexto del problema.

El proyecto se centra en el desarrollo de un modelo de aprendizaje automático para predecir los precios de vivienda en Colombia. Esta iniciativa surge en un entorno en el que los compradores y vendedores de propiedades inmobiliarias necesitan información precisa y confiable para tomar decisiones informadas en el mercado inmobiliario. Dado que los precios de las viviendas pueden fluctuar considerablemente en un país como Colombia, es crucial contar con un modelo predictivo que pueda proporcionar estimaciones precisas y oportunas de los precios de las propiedades en diferentes áreas geográficas.

El conjunto de datos recopilados incluye información detallada sobre una variedad de propiedades en diferentes departamentos y ciudades de Colombia, lo que proporciona una amplia cobertura geográfica. La recopilación de datos se llevó a cabo a través de un proveedor de datos y técnicas de scraping de diversas páginas web de venta de inmuebles.

La iniciativa también tiene como objetivo mantener la viabilidad a largo plazo del modelo, con la implementación de un plan para la recopilación periódica de datos que permitirá reentrenar el modelo y validar su precisión continua.

1.2. Pregunta de negocio y alcance del proyecto.

¿Cómo predecir con precisión los precios de vivienda en Colombia para facilitar a compradores y vendedores la toma de decisiones informadas en el mercado inmobiliario?

El alcance consiste en construir un modelo analítico que de acuerdo a las características del inmueble prediga el valor del precio del avalúo, y desarrollar un tablero de control que permita visualizar de manera interactiva un análisis exploratorio de la información de los inmuebles y una interfaz donde según la configuración del inmueble se pueda obtener el valor del mismo.

1.3. Breve descripción de conjuntos de datos a emplear.

Los datos a emplear corresponden a datos de 12.853 inmuebles de diferentes departamentos y ciudades de Colombia. El formato de los datos es un archivo de Excel donde se cuenta con toda la información de las propiedades y sus respectivos avalúos.

La recolección de datos se realizó a partir de un proveedor de datos con la información de diferentes proyectos de vivienda en Colombia cuya información se complementa utilizando técnicas de scraping de diferentes páginas web de venta de inmuebles en Colombia.

El objetivo de estos datos es estimar el precio total de un inmueble dadas unas características de este. En el conjunto de datos existe una columna llamada valor_total_avaluo, la cual es una variable numérica, que indica el precio total del inmueble que incluye áreas privadas, públicas, parqueaderos y cuarto útil en caso de tener alguno de estos.

1.4. Cambios con respecto a la primera entrega

Los cambios que realizamos respecto a la primera entrega se enfocan en el diseño inicial del tablero, ya que ahora contamos con otras visualizaciones y filtros, y para facilitar la navegación y aprovechar el espacio, se distribuyeron los componentes visuales de manera diferente.

2. Modelos desarrollados y su evaluación.

En el análisis exploratorio, observamos que varias de las variables categóricas tienen una asociación fuerte con la variable objetivo de precio, sin embargo, son variables de muchas categorías como por ejemplo el Departamento y el Municipio. Para poder utilizar este tipo de variables sin la necesidad de utilizar One Hot Encoding, debido a que generaría una explosión de variables dummies y entraríamos en el problema de la maldición del a dimensionalidad, decidimos codificar estas variables utilizando la librería Category Encoders. Es así que, previo a la creación de modelos, los datos se particionan en train y test y se aplica el encoder SummaryEncoder del módulo quantile_encoder de la biblioteca Category encoders. La clase SummaryEncoder es un codificador de categorías que utiliza la información de los cuantiles para transformar variables categóricas en representaciones numéricas. Tenemos en cuenta quantiles=(0.10, 0.25, 0.5, 0.75, 0.90), los cuantiles se utilizan para dividir la distribución de los datos en partes iguales, lo que permite asignar un valor numérico basado en la posición de la categoría en la distribución de los datos.

También teniendo en cuenta que tenemos una gran cantidad de variables luego de aplicar el encoder (293 variables), antes de entrenar modelos primero realizamos selección de variables utilizando la librería Featurewiz, cuyo objetivo principal es reducir el conjunto de variables a aquellas que realmente son relevantes para la predicción de la variable objetivo. Para nuestro análisis, se establece el límite de correlación en 0.90, es decir, que aquellas características con una correlación superior a 0.90 serán consideradas redundantes y eliminadas durante el proceso de selección de variables. Así pasamos de 293 a 103 variables.

Para el desarrollo de modelos utilizamos la librería PyCaret, que permite en una simple corrida comparar una amplia variedad de modelos. Como hablamos de un problema cuya variable objetivo es numérica (predicción de precios de vivienda), tuvimos en cuenta algoritmos de regresión, entre los cuales se comparan algoritmos basados en árboles como Random Forest, XGBoost, LigthGBM, entre otros, como también modelos basados en regresión lineal. PyCaret muestra diferentes métricas de evaluación como el error cuadrático medio (MSE), el error absoluto medio (MAE), el coeficiente de determinación (R-cuadrado) y Error Porcentual Absoluto Medio (MAPE). Para nuestro análisis la decisión de cuál es el mejor modelo es de acuerdo al MAPE.

A continuación puede observar las métricas que se obtuvieron para los diferentes algoritmos.

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
rf	Random Forest Regressor	36257653.2692	26366780633849048.0000	148522177.4104	0.6787	0.2181	0.1485	1.9880
xgboost	Extreme Gradient Boosting	38763691.2000	39929995378189928.0000	180529113.6000	0.4611	0.2146	0.1576	0.6600
et	Extra Trees Regressor	38795851.1611	35871284555126924.0000	167564564.7660	0.5404	0.2202	0.1578	1.7740
lightgbm	Light Gradient Boosting Machine	39049364.0505	24558265532933748.0000	141352422.0440	0.7471	0.2324	0.1624	0.1860
gbr	Gradient Boosting Regressor	42813214.5136	29346704400520556.0000	156198974.5883	0.6295	0.2448	0.1912	0.6680
dt	Decision Tree Regressor	53503836.6782	44601187227936944.0000	200077341.3751	0.3512	0.3051	0.2047	0.0800
huber	Huber Regressor	64234228.3792	33124458521091224.0000	178050197.2072	0.5749	0.4465	0.2474	0.3340
en	Elastic Net	71776425.6000	33281770725677464.0000	175978534.4000	0.5825	0.5954	0.3554	0.1260
omp	Orthogonal Matching Pursuit	75330064.0000	35162320438244148.0000	182775497.6000	0.5398	0.5213	0.3568	0.0380
br	Bayesian Ridge	75108673.6000	34675803078577356.0000	181022483.2000	0.5512	0.5278	0.3614	0.0400
lasso	Lasso Regression	72093411.2000	32714206259393332.0000	172995726.4000	0.6018	0.5905	0.3632	0.5960
llar	Lasso Least Angle Regression	76710236.8000	35401329140563968.0000	183212723.2000	0.5424	0.5445	0.3722	0.0460
lr	Linear Regression	82562564.8000	36666195076500688.0000	187462924.8000	0.5186	0.5836	0.4149	0.6400
ridge	Ridge Regression	86760004.8000	39262796153199000.0000	190831732.8000	0.5047	0.7146	0.4506	0.4440
par	Passive Aggressive Regressor	111975603.3802	37978206832555536.0000	191370353.6640	0.4083	0.6224	0.6320	0.0520
dummy	Dummy Regressor	152199440.0000	80598231854297904.0000	276143536.0000	-0.0007	0.7629	0.8971	0.0360
ada	AdaBoost Regressor	212158503.3305	87157570311540688.0000	288677601.7470	-0.2411	0.9615	1.6059	0.3720
lar	Least Angle Regression	88633479256934.4062	693279108190511733968857989 12.0000	126754431772726.4062	-439845074632.8664	6.6049	608142.9674	0.0400

3. Observaciones y conclusiones sobre los modelos.

1. Para la preparación de los datos y selección de variables, utilizamos la librería Featurewiz, la cual nos permitió reducir de 293 variables a 103 variables relevantes para la predicción de la variable objetivo que es el precio del avalúo del inmueble.

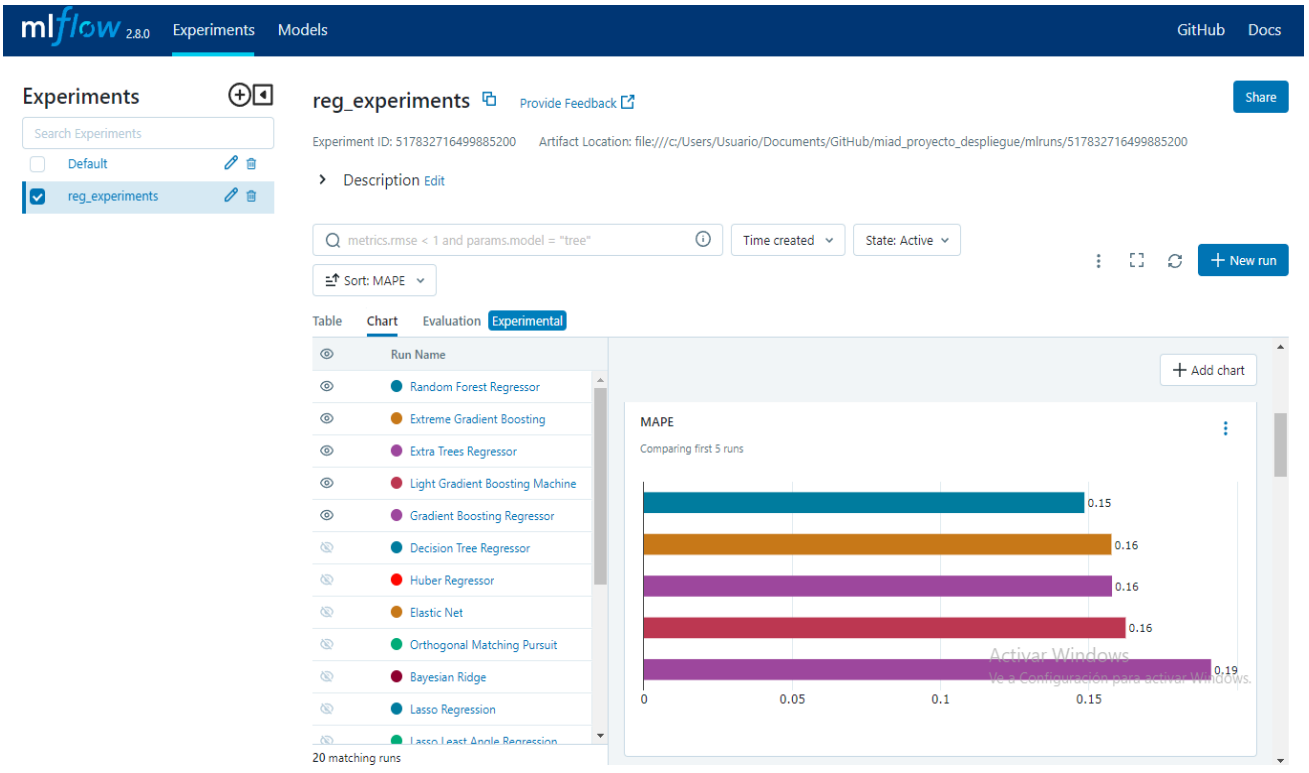
2. De acuerdo a la evaluación de diferentes modelos, la métrica de desempeño definida es el MAPE, que significa Error Porcentual Absoluto Medio (por sus siglas en inglés, Mean Absolute Percentage Error), la cual calcula el porcentaje promedio de la diferencia absoluta entre los valores predichos y los valores reales en relación con los valores reales, esta métrica es utilizada para evaluar la precisión de un modelo de regresión en pronósticos o predicciones. En los resultados del experimento con PyCaret, podemos observar que un modelo dummy (aleatorio) obtiene un MAPE de 0.89, mientras el mejor modelo (Random Forest) obtuvo un MAPE de 0.1485, el cual indica que el error promedio porcentual absoluto de las predicciones del mejor modelo es del 14.85%.

3. Aunque sigue siendo un MAPE algo alto, en general el MAPE es considerado bueno si está entre el 10% y 20%, y además, teniendo en cuenta que aún no hemos optimizado los algoritmos sino que estamos realizando comparaciones con sus configuraciones por defecto. En una próxima iteración optimizaremos los hiper parámetros del top 5 algoritmos que obtuvieron mejor resultado (Random Forest, XGBoost, Extra Trees, LightGBM y Gradient Boosting)

4. La ventaja de la métrica MAPE, es que se expresa como un porcentaje y es útil para comprender el error relativo en las predicciones de un modelo, es fácil de interpretar y comparar entre diferentes conjuntos de datos y modelos.

5. Posibles limitaciones al usar MAPE como métrica de desempeño es que puede tener problemas al evaluar valores de referencia cercanos a cero, lo que puede resultar en una divergencia o valores infinitos. Por lo tanto, es importante considerar el contexto, datos, variables a la hora de su interpretación.

6. PyCaret, además de generar el reporte de las métricas, también registra el experimento en MLFlow, donde podemos observar que el top modelos según MAPE corresponden con los listados en el notebook por PyCaret.



Top variables según importancia de variables del mejor modelo

Importancia de variables

```
importances = model.steps[2][1].feature_importances_
feature_names = X_test_selected.drop(columns=["clean_valor_total_avaluo"]).columns
forest_importances = pd.DataFrame(importances, index=feature_names, columns=['importance']).reset_index()
forest_importances['orig_var'] = forest_importances['index'].apply(lambda v: vars_dict[v])
forest_importances.groupby(['orig_var'])['importance'].max().sort_values(ascending=False).head(30).index
```

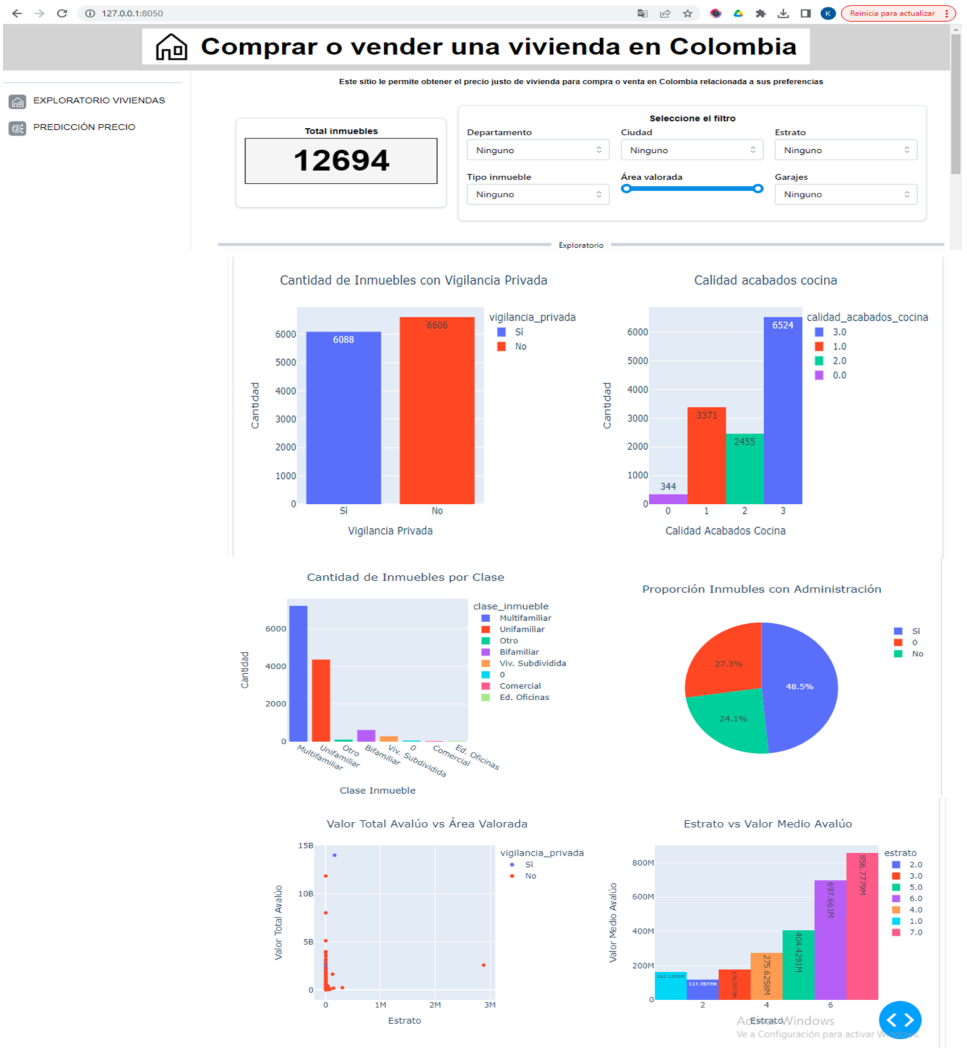
```
Index(['barrio', 'area_valorada', 'vetustez', 'ocupante',
       'total_cupos_parquedaro', 'municipio_inmueble', 'estrato', 'cocina',
       'clase_inmueble', 'numero_piso', 'estructura_reforzada',
       'departamento_inmueble', 'tipo_garaje', 'detalle_material',
       'tipo_inmueble', 'closet', 'bano_privado', 'balcon',
       'calidad_acabados_madera', 'tanque_de_agua', 'terraza',
       'calidad_acabados_metal', 'demanda_interes', 'telefono_en_el_predio',
       'gas_en_el_predio', 'calidad_acabados_cocina', 'numero_de_edificios',
       'calidad_acabados_techos', 'tipo_deposito', 'motivo'],
      dtype='object', name='orig_var')
```

3. Descripción del tablero desarrollado y la funcionalidad que éste ofrece.

El tablero está implementado en el framework Dash de Python y consta de dos módulos: "Exploración de Viviendas" y "Predicción de Precio".

Exploración de Viviendas:

El objetivo de este módulo es proporcionar a los usuarios información relevante para que puedan interactuar con los datos de viviendas a través de varios filtros (municipio, tipos de vivienda, área valorada, etc.). Permitiendo a los usuarios ver la influencia de diferentes variables en el precio y características de las viviendas.



Predicción de Precio:

En la sección de predicción de precios, los usuarios tienen la opción de configurar varios criterios para predecir el precio de una vivienda. Estos criterios incluyen la ubicación (departamento, ciudad, barrio), estrato de la vivienda, área valorada y características específicas de los inmuebles como el número de habitaciones, baños, estacionamiento, etc. Adicional, se incluyó la función de hacer predicciones de forma aleatoria.

- **Repositorios con todo el código (accesibles y con evidencia sobre el uso de repositorios por parte de los miembros del equipo).**

Repositorio Git en uso para el código: https://github.com/asernat/miad_proyecto_despliegue

Repositorio DVC en uso para los datos: El repositorio DVC para los datos se realizó en Google Drive en la siguiente carpeta:
https://drive.google.com/drive/folders/1HyhWYmunC9PaANnc6PlHug9vJFGUKrrA

- **Fuentes de los modelos desarrollados**.

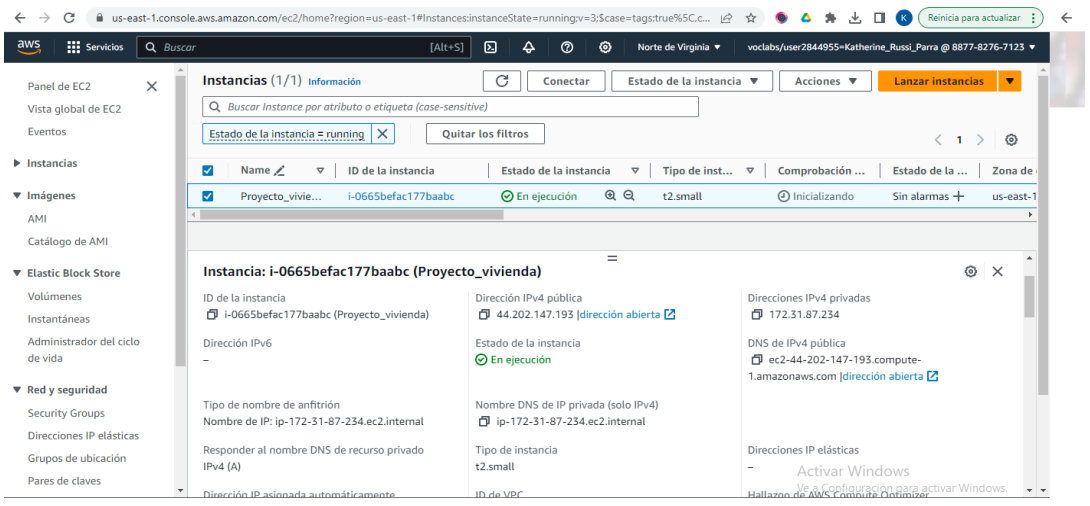
https://github.com/asernat/miad_proyecto_despliegue/blob/master/codigo/05%20-%20Modelos.ipynb

- **Fuentes del tablero desarrollado.**

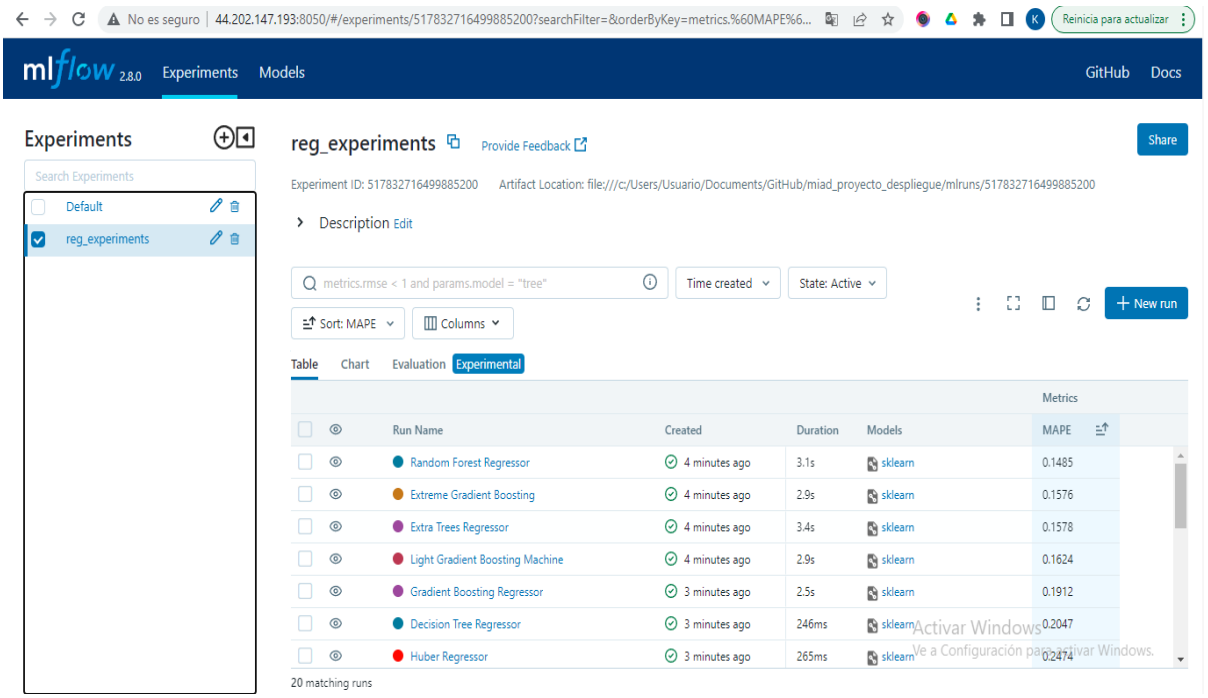
https://github.com/asernat/miad_proyecto_despliegue/tree/master/app

- Pantallazos de experimentos registrados en MLflow en una máquina de AWS EC2 (debe ser visible el usuario e IP de la máquina en EC2, y la IP en MLflow). Mantenga su máquina en EC2 con MLflow detenida (no la termine).

Creamos una máquina en AWS EC2 con Ubuntu server, instancia small con 20 GB de disco.



Los experimentos de MLflow son:



- **Reporte de trabajo en equipo de máximo 1 página.**

Nombre	**Responsabilidades**	**Commits**
Todos	- Problema que abordarán y su contexto - Pregunta de negocio y alcance del proyecto. - Ajustes de visualización del tablero	
Andres Santiago Serna Tangarife	- Descripción conjunto de datos - Construcción tablero en Dash - Revisión, conexión y ajuste de cálculo predicción.	- modelos selección variables - actualización tablero
Daniela Castillo	- Modelos desarrollados y su evaluación	- correr modelos
William Morales	- Descripción del tablero desarrollado - test de modelo desarrollados	- Ajuste gráficas tablero - testeo de los modelos
Katherine Russi Parra	- Observaciones y conclusiones de los modelos	- Resultado experimento MLFlow - plan y primera entrega del proyecto