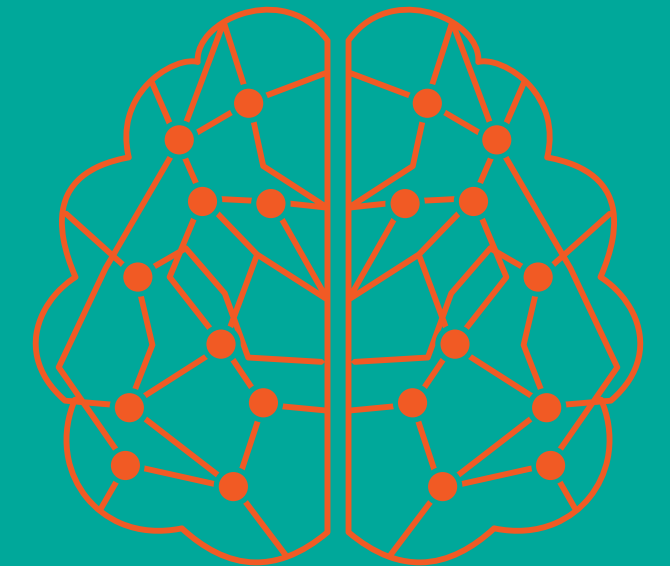


INTELIGENCIA ARTIFICIAL

Pollair

Sistema de predicción de agentes
contaminantes en el aire en
Medellín-Colombia

Erika Yamile Lache Blanco
Código: 2140348



MOTIVACIÓN







¿Por qué? ¿Para qué?

- 1 Conocer parámetro $pm_{2.5}$ contaminante en el aire.
- 2 Tomar medidas preventivas de movilidad.
- 3 A largo plazo, Evitar enfermedades pulmonares

Objetivos



-  Predecir correctamente el valor del parámetro teniendo en cuenta teniendo en cuenta las seis horas registradas anterior a este.
-  Cargar, tratar y almacenar óptimamente el dataset.
-  Proponer alternativas que puedan generar mejoras en el resultado final del proyecto.
-  Elegir entre las alternativas el método de mayor precisión en la regresión.

TRATAMIENTO DEL DATASET

🌸 Total de datos en el dataset = 90345

🌸 Datos con el parámetro $Pm_{2.5}$ = 5.4677

🌸 Datos con idLocalidad 7063 = 4.458

🌸 Valor de elementos nulos (-9.9999) = 236

🌸 Dataset resultante = 4.222

🌸 Total de valores = 4.216

Recordatorio

Limpieza realizada al dataset para poder implementar los métodos de regresión.

CLASIFICADORES

Decision Tree

modelo que prediga el valor de una variable objetivo en función de varias variables de entrada.

Random Forest

Creado para modelos de clasificación o regresión mediante la construcción de árboles de decisión. Es una de los algoritmos más flexibles y fáciles de implementar.

SVR

Es el tipo de algoritmo de aprendizaje profundo que realiza el aprendizaje supervisado para clasificación o regresión de grupos de datos.

Implementación

DTR

```
regressor_DTR = DecisionTreeRegressor()  
regressor_DTR.fit(X_train_DTR, y_train_DTR)
```

```
regressor_RFR = RandomForestRegressor()  
regressor_RFR.fit(X_train_RFR, y_train_RFR)
```

RFR

SVC

```
est_SVR = SVR(kernel=kernel)  
est_SVR.fit(X_train_SVR, y_train_SVR)
```

RESULTADOS

Errores obtenidos con la implementación del modelo.

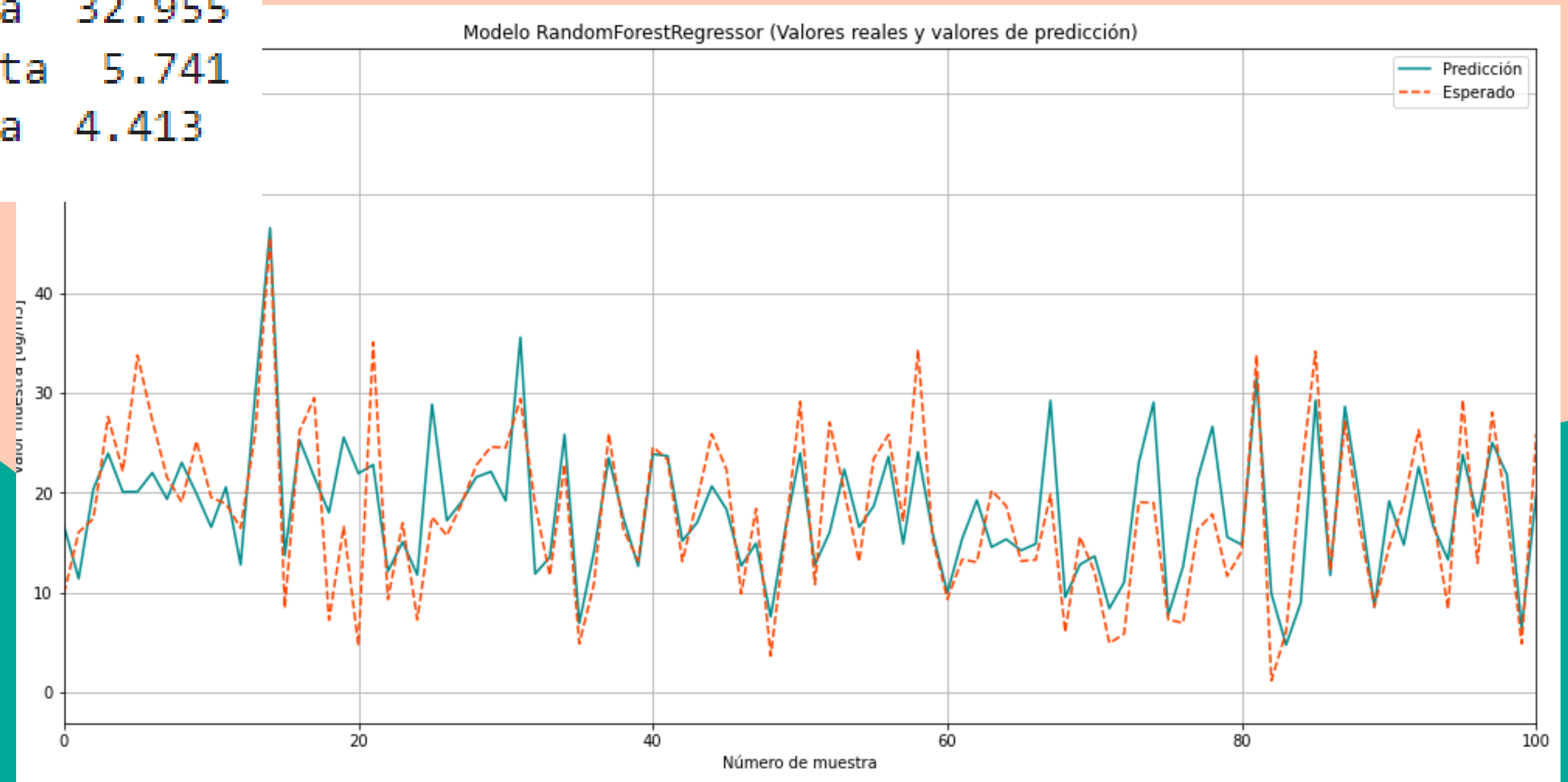
```
MSE depth split data 32.955
RMSE depth split data 5.741
MAE depth split data 4.413
```



RESULTADOS

Errores obtenidos con la implementación del modelo.

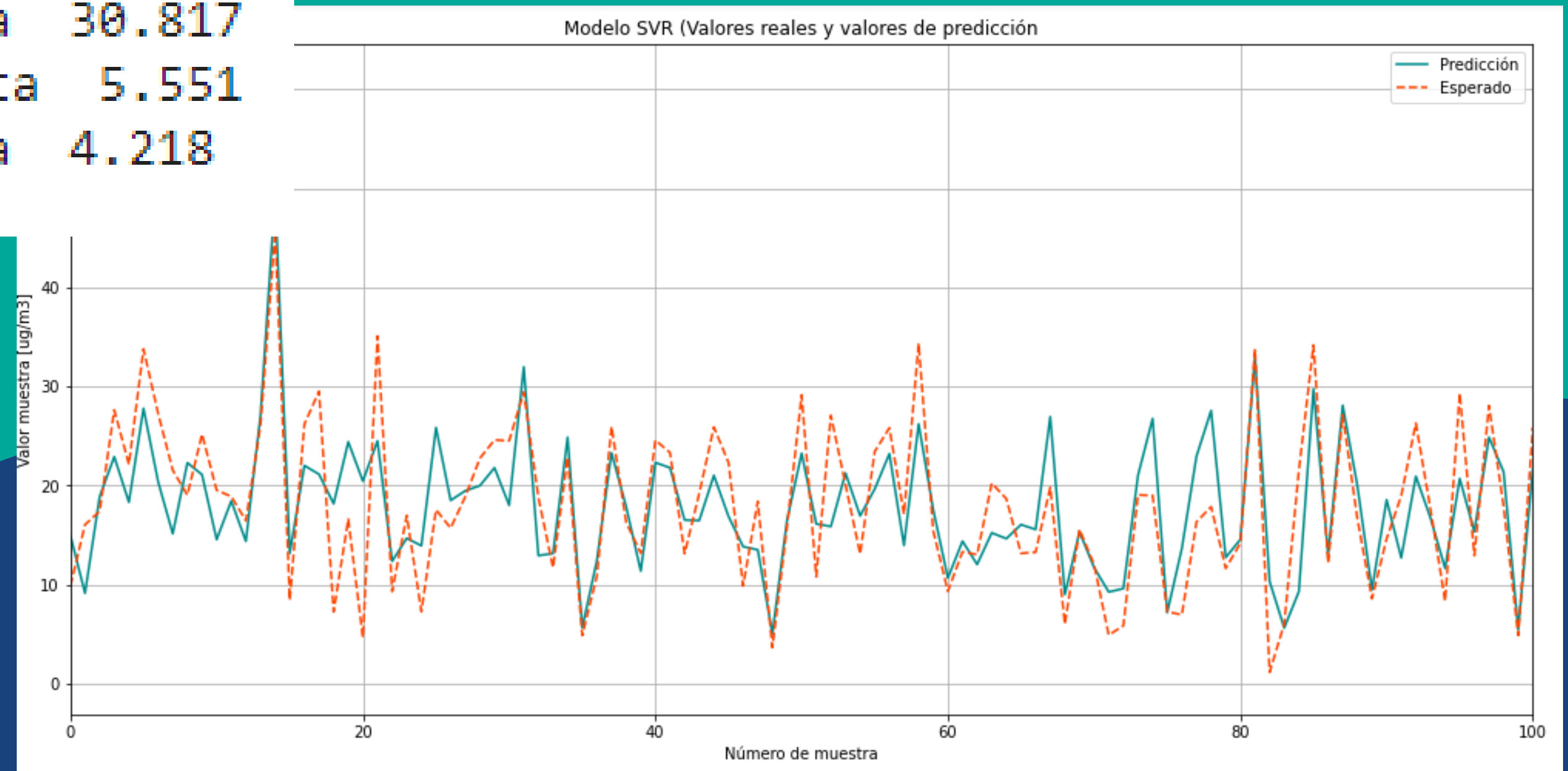
```
MSE depth split data 32.955
RMSE depth split data 5.741
MAE depth split data 4.413
```



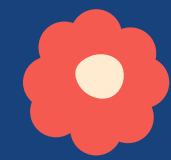
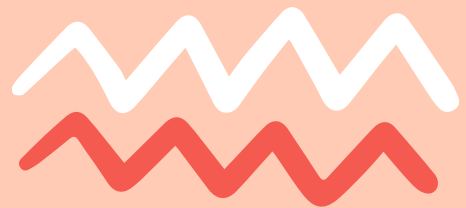
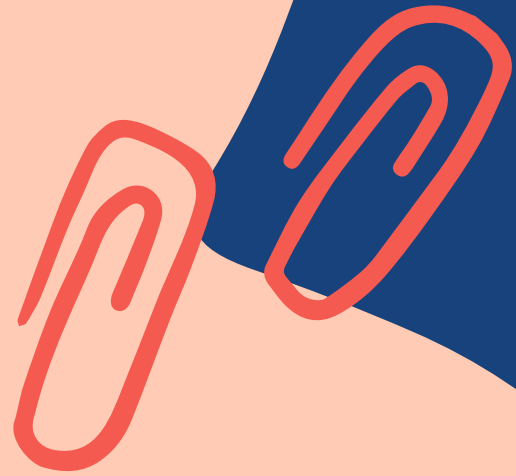
RESULTADOS

Errores obtenidos con la implementación del modelo.

```
MSE depth split data 30.817
RMSE depth split data 5.551
MAE depth split data 4.218
```



Red Neuronal ✨



Son una serie de algoritmos que imitan las operaciones de un cerebro humano para reconocer las relaciones entre grandes cantidades de datos

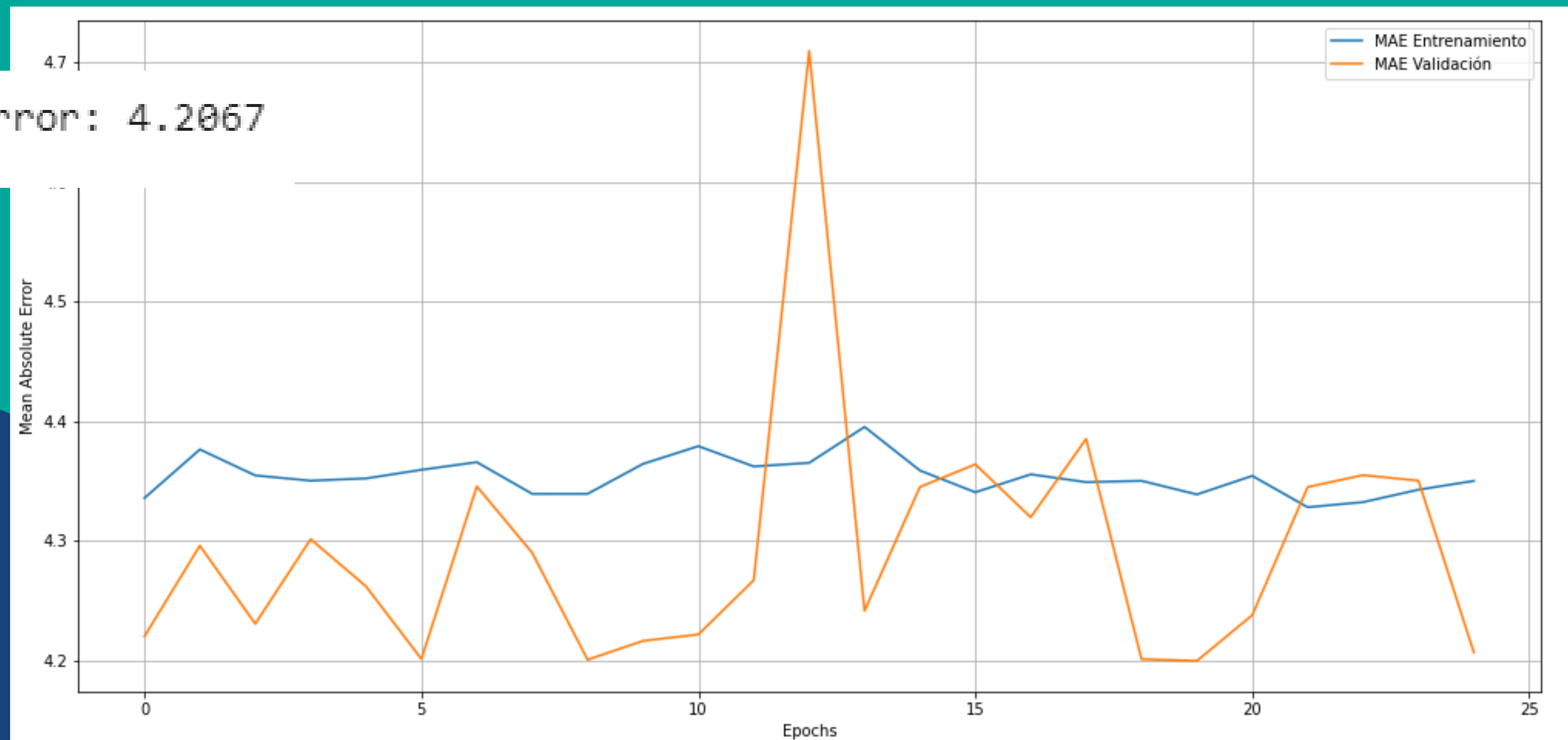
Implementación

```
model_DNN = tf.keras.Sequential([  
    tf.keras.layers.Flatten(input_shape = X_train_DNN[0].shape),  
    tf.keras.layers.Dense(3, activation=tf.nn.relu),  
    tf.keras.layers.Dense(1, activation=tf.nn.relu)  
])
```

RESULTADOS

Errores obtenidos con la implementación del modelo.

```
val_mean_absolute_error: 4.2067
```



ERRORES

lll

de cada método

	MSE	RMSE	MAE
DTR	32.955	5.741	4.413
RFR	32.955	5.741	4.413
SUR	30.817	5.551	4.218
RN	-	-	4.206

