



Equipo 3

SISTEMA PROACTIVO DE APOYO A LA TOMA DE DECISIONES INTELIGENTE (IDSS)

Fase 2 /Avance del Proyecto

Manuel Antonio Acevedo Ham	A01410910
Jorge Daniel Amezola Gutiérrez	A01793759
Ana Clemencia Aristizábal Londoño	A01795433
Alan Samael Arriaga Castillo	A01153355
Kevin Balderas Sánchez	A01795149



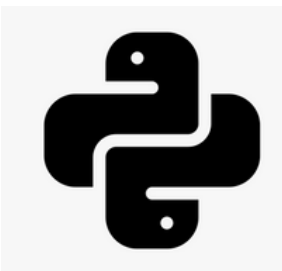
Acercamiento inicial del problema

Para la correcta implementación de la solución, hemos utilizado herramientas de colaboración que permiten una integración del proyecto final, logrando evaluar en todo momento la ruta crítica;



GitHub

<https://github.com/A01795433-AnaAristizabal/MLOPSGrupo3>



Python



Data Version Control



Documentos y
Videos

https://drive.google.com/drive/folders/1XgbRPmMgcQyO6gszxAiOtmi5G8fMYO_6?usp=sharing



Project template para Python



VS Code



Object Server



Jupyter Notebooks








Experimentos ML

Actividades por Rol en Cada Fase

Durante la aplicación de la metodología MLOps (Machine Learning Operations) para la elaboración del ejercicio, cada integrante del equipo ha asumido un rol con un conjunto específico de actividades para el desarrollo e implementación de los modelos de machine learning que luego serán llevados a producción.

- ✓ Analista de datos, ha proporcionado la base de datos que ha impulsado el modelo de machine learning (ML). Su trabajo de preparación y limpieza de datos ha asegurado que la Data Scientists Ana comenzará la construcción de los modelos de ML sobre datos fiables y de calidad.
- ✓ La Data Scientists, ha creado y experimentado con los algunos modelos (son la base del ciclo de vida de machine learning en MLOps) Su trabajo está alineado con las fases de entrenamiento y evaluación dentro del ciclo. en las siguientes fases del ejercicio trabajará en estrecha colaboración con el MLOps Engineer Daniel para trasladar los modelos entrenados a producción de forma eficiente y repetible.
- ✓ ML Engineer se ha ocupado de convertir el modelo desarrollado por la Data Scientists Ana en un sistema que para la siguientes fases se espera que se más robusto, reproducible y escalable, automatizando pipelines y en la integración de los modelos dentro del flujo de producción
- ✓ Software Engineer y MLOps Engineer asegurarán que los modelos se integren en sistemas de software y que puedan ser desplegados de manera segura y escalable

Actividades por Rol en cada fase

	Fase 1	Fase 2	Fase 3	Fase 4
 Ana	Data Scientist	ML Engineer		
 Alan	Software Engineer	MLOps Engineer		
 Daniel	MLOps Engineer	Data Analyst		
 Revin	Data Analyst	Data Scientist		
 Manuel	ML Engineer	Software Engineer		

Análisis del Problema

Problema a Resolver

El principal objetivo de este análisis es predecir la cantidad de veces que se compartirá un artículo en las redes sociales (shares) identificando los factores clave que influyen en la popularidad de los artículos publicados por Mashable.

La tarea principal es el aprendizaje supervisado, se trata de un problema de regresión, en el que el objetivo, es predecir el número de “shares” de los artículos.

Resolver este problema para optimizar el contenido publicado para maximizar el alcance y la visibilidad de los artículos en línea.

Descripción del Conjunto de Datos

El conjunto de datos contiene información sobre artículos publicados en el sitio web de Mashable durante dos años.

Puntos clave:

Período: dos años.

Variable objetivo: la cantidad de veces que se compartió un artículo en las redes sociales (popularidad).

Número de registros: 39 644 artículos.

Número de características: 61 atributos.

Características: Valores numéricos -Valores booleanos - Valores categóricos
Los datos incluyen una variedad de características relacionadas con los artículos, como su categoría, fecha de publicación, duración, y métricas de popularidad

Variable objetivo: Cantidad de veces que se compartió el artículo, que se puede usar como una tarea de regresión o clasificación según cómo se plantee el problema

Métodos y Técnicas Utilizadas Para el Análisis del Problema

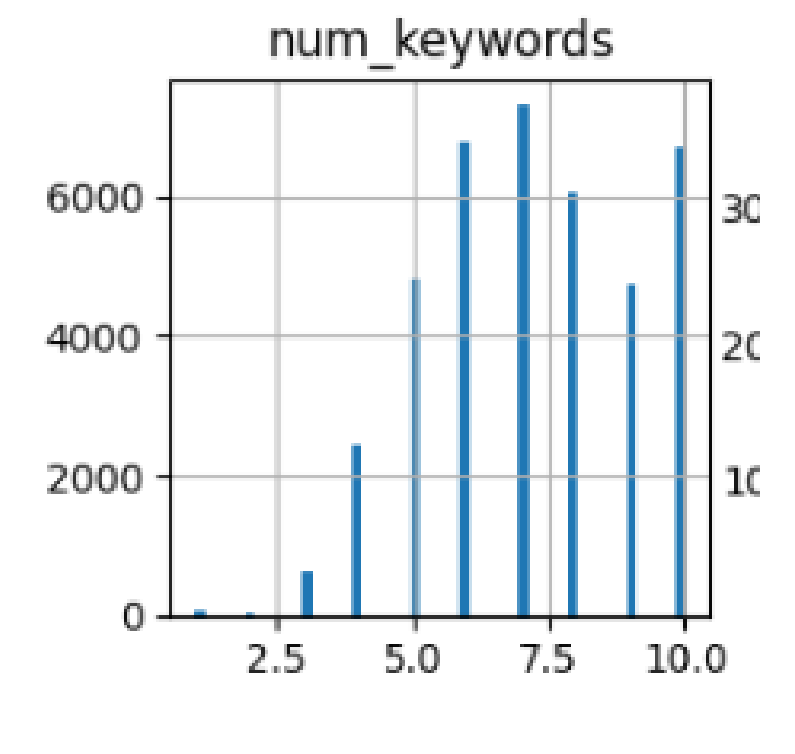
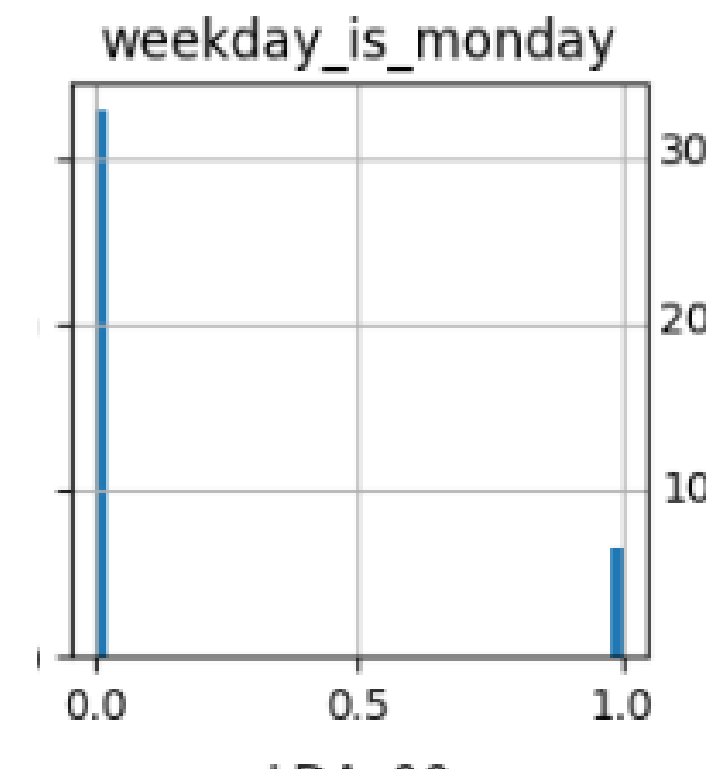
Exploración y preprocesamiento de los datos

1. Verificar valores faltantes: Nuestro conjunto de datos no tiene valores faltantes, por lo que no fue necesario realizar ningún proceso de imputación

2. Análisis de la distribución de los datos: Al evaluar la distribución de los valores en las diferentes columnas se pudo observar diferentes escalas, algunas características no tienen los valores necesarios para ver generar un histograma, por ejemplo `week_is_monday` aunque es un valor numérico, es una característica binaria, pero otras como `num_keywords` tienen pocas categorías, por lo que en el histograma no es posible ver una curva bien generada.

```
online_news_popularity_df.info() #Checking for nulls or missing data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39644 entries, 0 to 39643
Data columns (total 59 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   n_tokens_title                        39644 non-null  float64
1   n_tokens_content                      39644 non-null  float64
2   n_unique_tokens                      39644 non-null  float64
3   n_non_stop_words                    39644 non-null  float64
4   n_non_stop_unique_tokens            39644 non-null  float64
5   num_hrefs                          39644 non-null  float64
6   num_self_hrefs                     39644 non-null  float64
7   num_imgs                           39644 non-null  float64
8   num_videos                         39644 non-null  float64
9   average_token_length                39644 non-null  float64
10  num_keywords                        39644 non-null  float64
11  data_channel_is_lifestyle            39644 non-null  float64
12  data_channel_is_entertainment        39644 non-null  float64
13  data_channel_is_bus                  39644 non-null  float64
14  data_channel_is_socmed              39644 non-null  float64
```

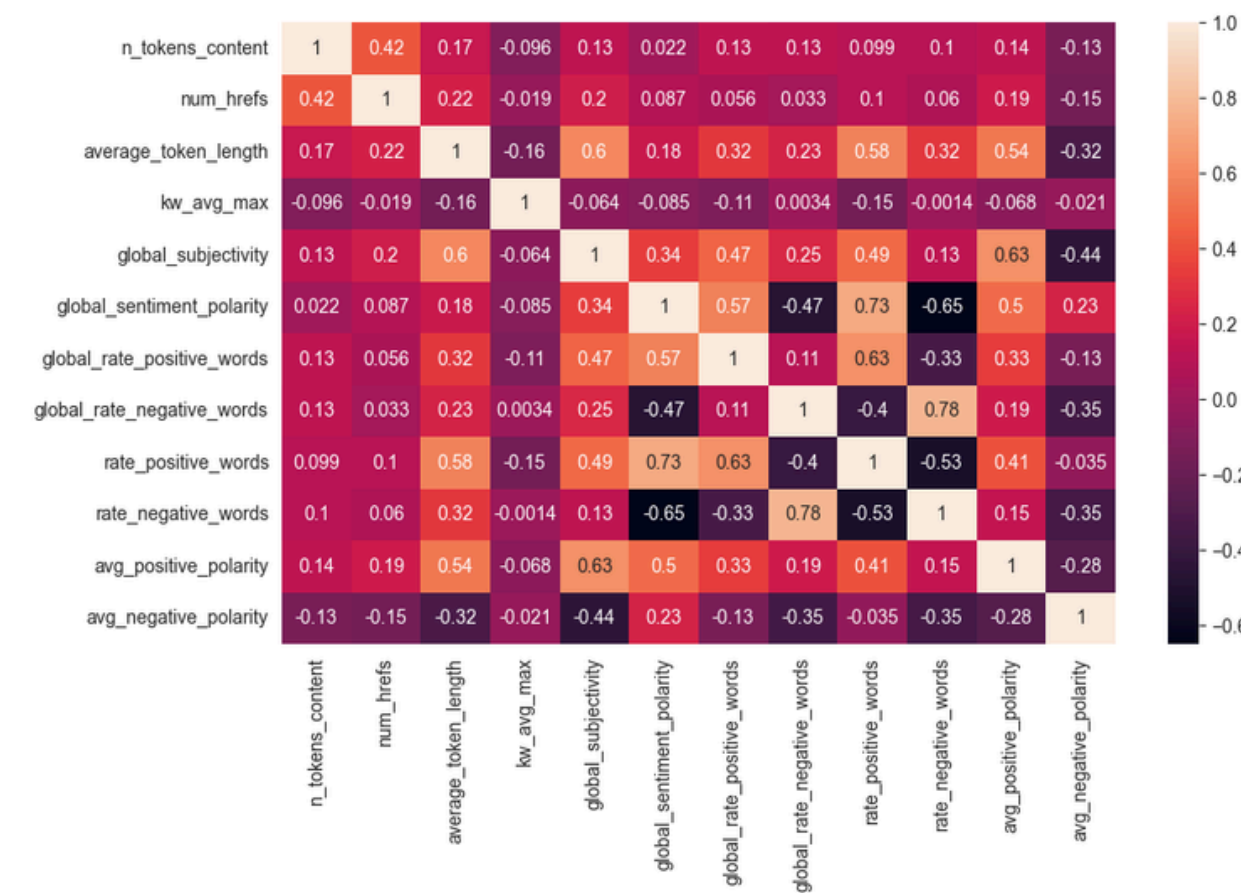
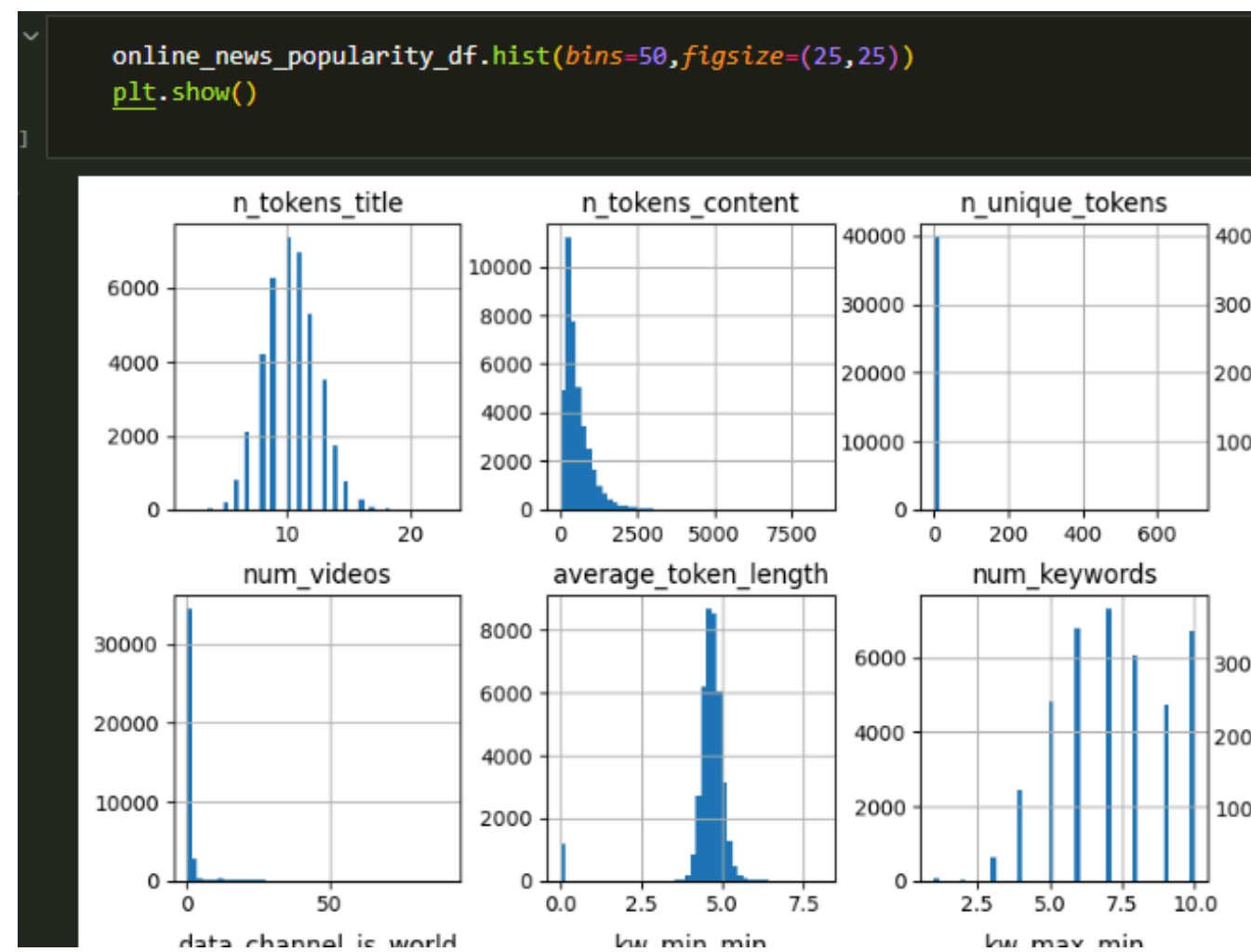


Métodos y Técnicas Utilizadas Para el Análisis del Problema

Exploración y preprocesamiento de los datos

3. Análisis descriptivo: utilizando histogramas para ver el comportamiento y distribución. Algunas variables tienen una distribución centrada y varias con sesgo hacia la izquierda y derecha y mapas de calor.

También hicimos uso del coeficiente de correlación de person para analizar la relación de la variable objetivo shares, con las demás variables



Métodos y Técnicas Utilizadas Para el Análisis del Problema

Exploración y preprocesamiento de los datos

3. EDA y Transformación de variables: Algunas variables requirieron transformaciones como raíz cuadrada, logaritmo, potencia2 y yeo-johnson para tratar las distribuciones sesgadas para facilitar el análisis posterior. La tarea principal es el aprendizaje supervisado, se trata de un problema de regresión, en el que el objetivo, es predecir el número de “shares” de los artículos

```
##Preprocessing and feature engineering

def transformaciones_1(data):
    print("*****Transforma
    cols_names_numeric= columns_numeric()
    variables_a_transformar = cols_names_numeric[:11]
    n = len(variables_a_transformar)
    misdatos = data

    sbn.set(rc={'figure.figsize':(17,12)})
    fig, axes = plt.subplots(5, n)

    for k in range(0,n):
        # Datos originales -----
        plt.subplot(5,n,k+1+(n*0))

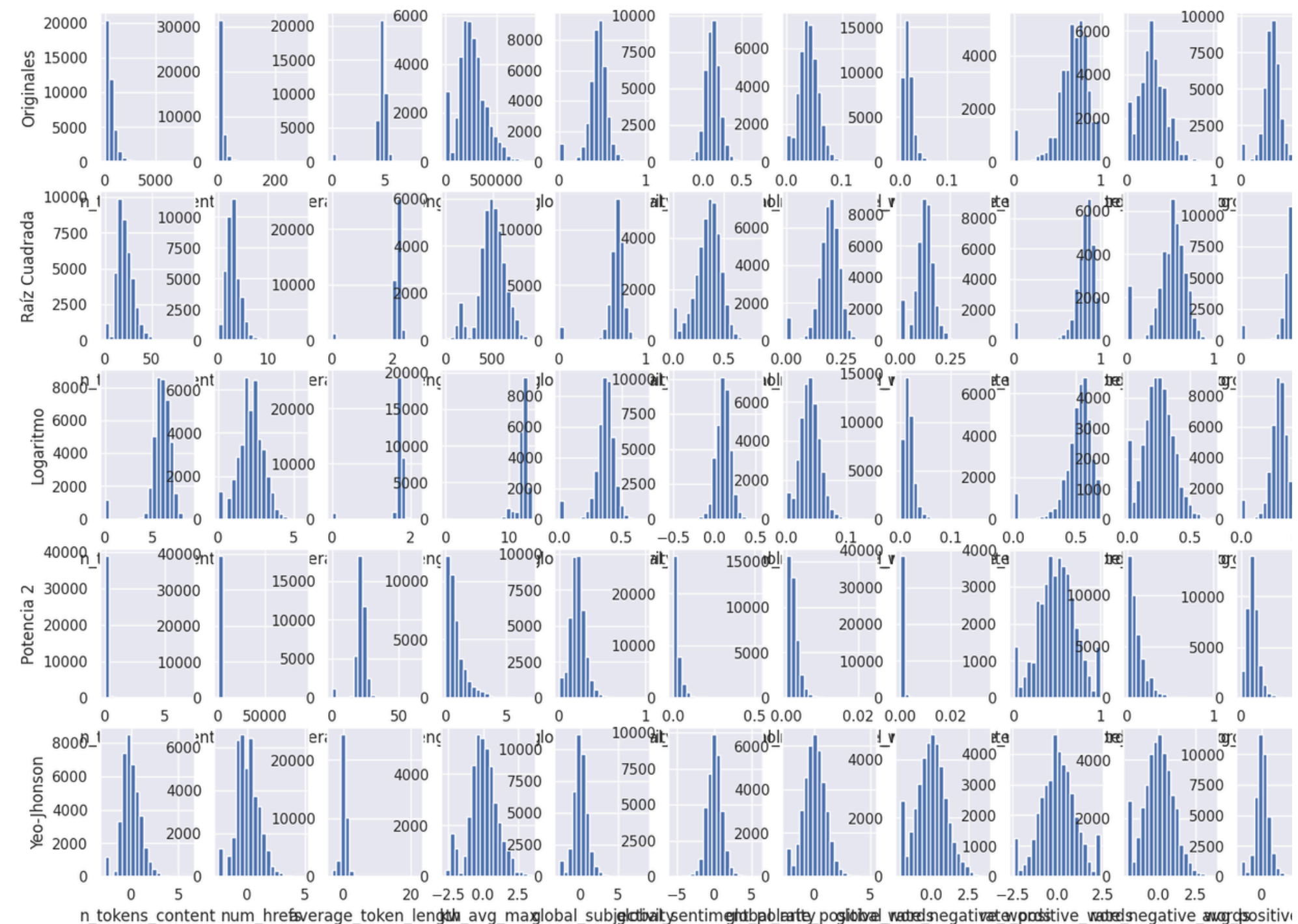
        Transf0 = misdatos[variables_a_transformar[k]] # En esta l
        plt.hist(Transf0,bins=20) # En este línea agrega el comand
        plt.xlabel(variables_a_transformar[k])
        if k==0:
            plt.ylabel('Originales')

        # Datos transformados con raíz cuadrada -----
        plt.subplot(5,n,k+1+(n*1))

        Transf1 = np.sqrt(misdatos[variables_a_transformar[k]])
        plt.hist(Transf1,bins=20) # En este línea
        plt.xlabel(variables_a_transformar[k])
        if k==0:
            plt.ylabel('Raíz Cuadrada')

        # Datos transformados con logaritmo natural -----
        plt.subplot(5,n,k+1+(n*2))

        Transf2 = np.log1p(misdatos[variables_a_transformar[k]])
        plt.hist(Transf2,bins=20) # En este línea
        plt.xlabel(variables_a_transformar[k])
        if k==0:
            plt.ylabel('Logaritmo')
```



El modelo

Metodología de Entrenamiento:

Para profundizar en la relación entre las características de los artículos y su popularidad decidimos aplicar un enfoque de modelado predictivo. Empleamos modelos supervisados, como árboles de decisión y un bosque aleatorio para predecir la popularidad de los artículos basada en las características disponibles en el conjunto de datos

DecisionTreeRegressor

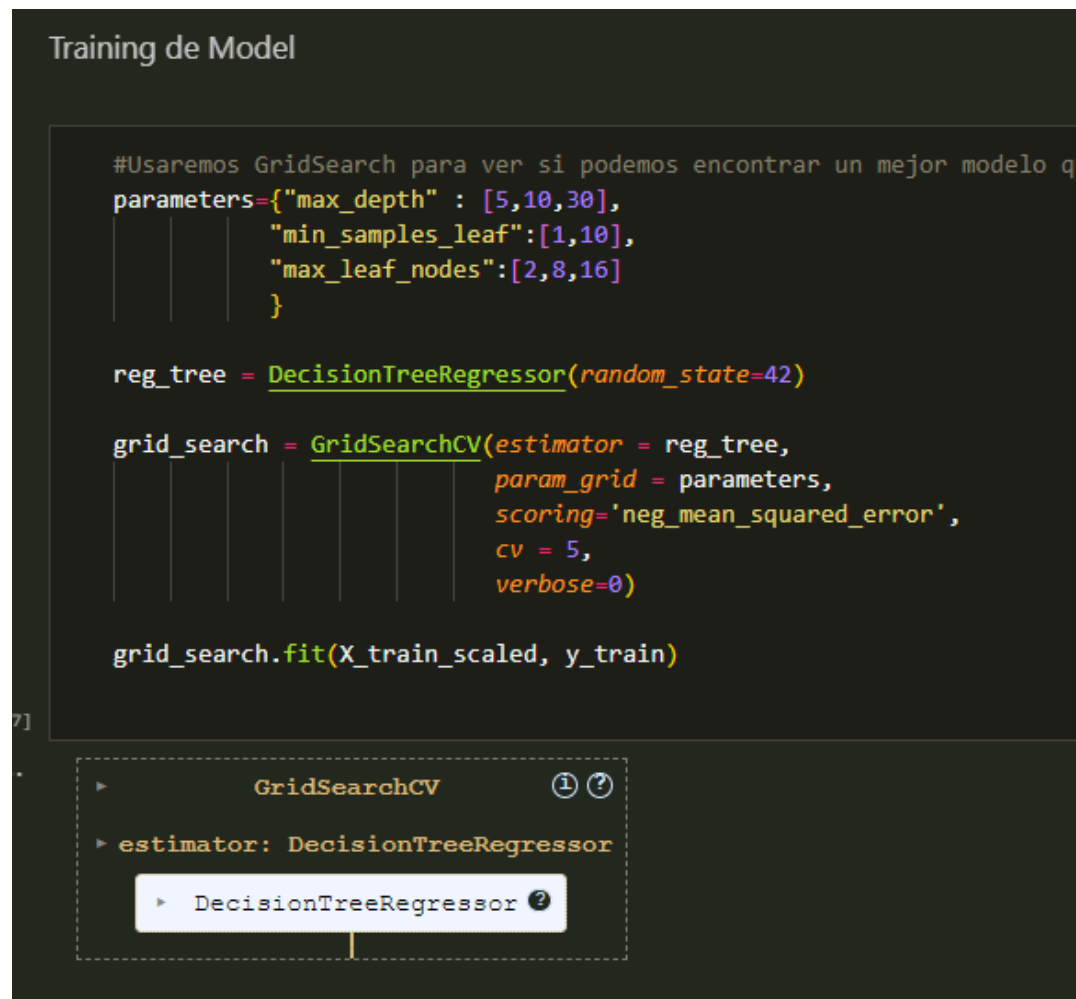
```
Training de Model

#Usaremos GridSearch para ver si podemos encontrar un mejor modelo que
parameters={"max_depth" : [5,10,30],
            "min_samples_leaf": [1,10],
            "max_leaf_nodes": [2,8,16]
            }

reg_tree = DecisionTreeRegressor(random_state=42)

grid_search = GridSearchCV(estimator = reg_tree,
                           param_grid = parameters,
                           scoring='neg_mean_squared_error',
                           cv = 5,
                           verbose=0)

grid_search.fit(X_train_scaled, y_train)
```



RandomForestRegressor

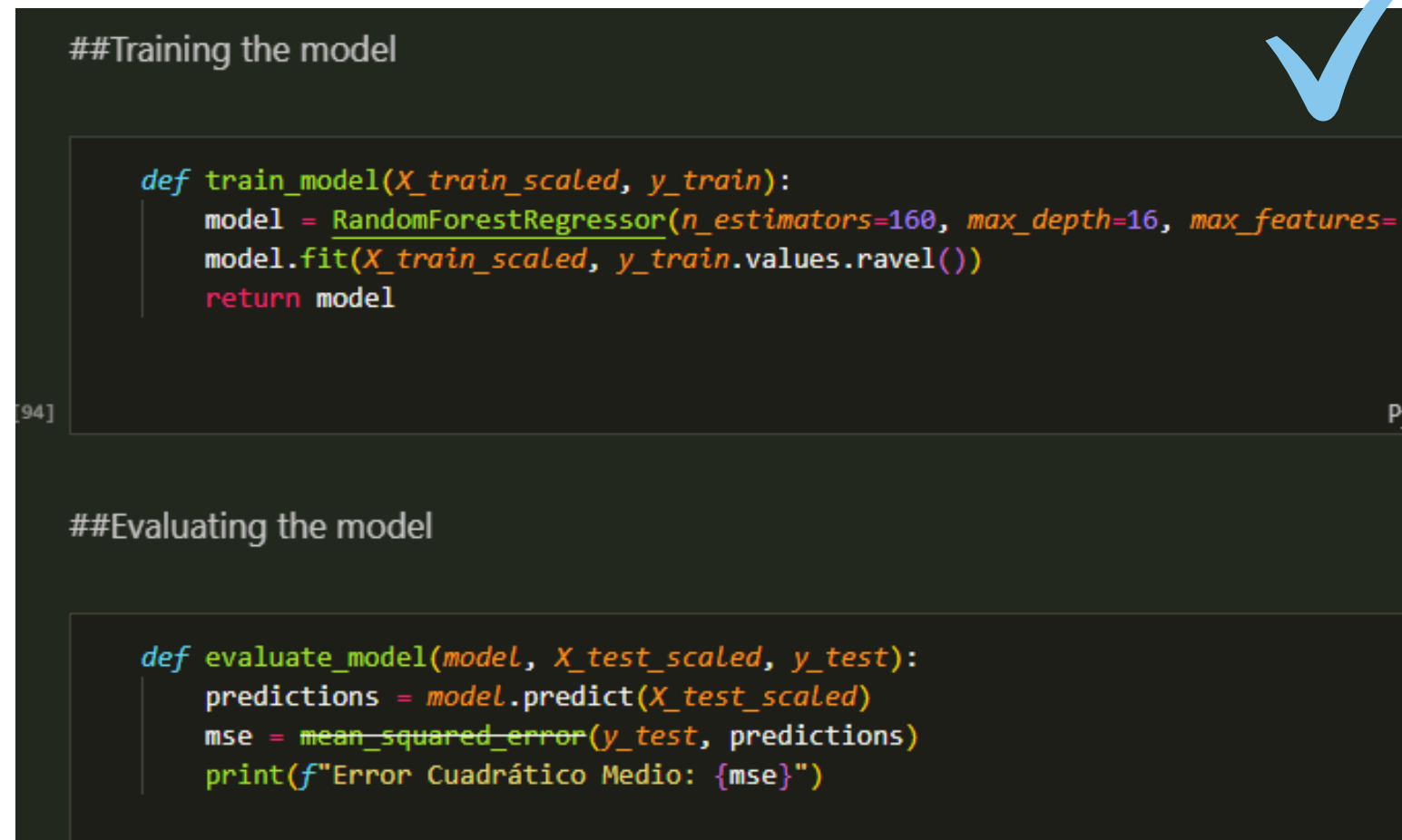
```
##Training the model

def train_model(X_train_scaled, y_train):
    model = RandomForestRegressor(n_estimators=160, max_depth=16, max_features='sqrt')
    model.fit(X_train_scaled, y_train.values.ravel())
    return model

[94]

##Evaluating the model

def evaluate_model(model, X_test_scaled, y_test):
    predictions = model.predict(X_test_scaled)
    mse = mean_squared_error(y_test, predictions)
    print(f"Error Cuadrático Medio: {mse}")
```



Después de evaluar ambos modelos, optamos por el modelo Random Forest

El modelo

Análisis de resultados del modelo y métricas utilizadas

Áreas a mejorar en el análisis del caso

Después de aplicar la metodología ML, evaluamos los modelos mediante la métricas de: Error Cuadrático Medio (MSE)

```
##Evaluating the model

def evaluate_model(model, X_test_scaled, y_test):
    predictions = model.predict(X_test_scaled)
    mse = mean_squared_error(y_test, predictions)
    print(f"Error Cuadrático Medio: {mse}")

[5]
```

Error Cuadrático Medio (MSE)

```
tree_Reg = RandomForestRegressor()
tree_Reg.fit(X_train_scaled, y_train)

preds = tree_Reg.predict(X_train_scaled)
mse = mean_squared_error(y_train, preds)
print(f"Error Cuadrático Medio: {mse}")

[16]
.. Error Cuadrático Medio: 0.22702768615784408

preds = tree_Reg.predict(X_test_scaled)
mse = mean_squared_error(y_test, preds)
print(f"Error Cuadrático Medio: {mse}")

[17]
.. Error Cuadrático Medio: 1.6618155539436172
```

Al momento de entrenar el modelo obtuvimos un 0.22

Al momento de poner el modelo a prueba con el conjunto de datos test obtuvimos un valor de 1.6

MLOps Refactorización en Clases

Como tarea de esta fase dos, se tuvo que refactorizar código en clases para tener mayor modularidad y escalabilidad. La clase Data Explorer hace exploración de datos y visualización. Luego la clase OnlineSharesModel hace el training del modelo y toda la etapa que esto implica

Se decidió dividir en una clase DataExplorer y otra OnlineSharesModel

Executing code

```
In [76]: filepath=r'C:/Users/balde/Desktop/MAESTRIA MNA/Contribs/MLOPSGrupo3/data/raw/online_news_popularity.csv'

model = OnlineShares(filepath)
model.load_data()
X_train_scaled, X_test_scaled, y_train, y_test = model.preprocess_data()
print()
print('Training-----')
model_reg = model.train_model(X_train_scaled, y_train)
print(model_reg)
print()
print('Testing-----')
model.evaluate_model(model_reg, X_test_scaled, y_test)
print()
print('Cross Validation -----')
model.cross_validate_model()

avg_negative_polarity-Transformed      avg_negative_polarity-Original

Puntos a destacar, podemos aplicar las siguientes transformaciones a las siguientes variables para corregir skew:
Númericas - Standard Scaler para que todos los rangos esten entre 0 y 1
num_hrefs - n_tokens_content -avg_positive_polarity - global_subjectivity - Logaritmo - np.log1p
average_token_length - global_sentiment_polarity - Original
kw_avg_max - global_rate_positive_words- Raiz Cuadrada
global_rate_negative_words - rate_positive_words -rate_negative_words- Yeo Johnson

Training-----
RandomForestRegressor(max_depth=16, max_features='sqrt', min_samples_leaf=2,
                      n_estimators=160, random_state=42)

Testing-----
Error Cuadrático Medio: 1.4825105532379537

Cross Validation -----
Average Accuracy with CV: 0.15668938377768532

Out[76]: <__main__.OnlineShares at 0x2e2a7cb85f0>
```


MLOps Refactorización en Clases

Data Explorer

```
74]: class DataExplorer:
    @staticmethod
    def explore_data(data):
        print(data.head().T)
        print(data.describe())
        print(data.info())

    @staticmethod
    def clear(data):
        #We can see there are a few records with namespacing between the column name, so we need to make a trim
        columns_clean = [i.strip() for i in data.columns.values.tolist()]
        data.columns = columns_clean

    @staticmethod
    def basic_vis(data):

        data['shares'].values
        data["n_tokens_content"].plot(kind = "box")
        plt.show()
        data.mean(axis=0)
        data['n_tokens_title'].plot(kind = "hist")
        plt.show()
        print("*****skew*****")
        data.skew()
        print("*****kurtosis*****")
        data.kurtosis()

    @staticmethod
    def plot_histograms(data):
        data.hist(bins=50, figsize=(25, 25))
        plt.show()

        insight1="A couple of comments: Different scales, some features do not have the necessary values to see gener
        insight2="for example week_is_monday although is a numeric value, is a binary feature , but others like num_key
        insight3="so the histogram is not possible to see a well-generated curve For a first model, we'll take into ac
        insight4="n_tokens_content num_hrefs average_token_length kw_avg_max global_subjectivity , global_sentiment_po
        insight5="rate_positive_words , rate_negative_words avg_positive_polarity , avg_negative_polarity Binary featu
        insight6="data_channel_is_bus, data_channel_is_socmed data_channel_is_tech, data_channel_is_world , weekday_is
        insight7="weekday_is_thursday, weekday_is_friday, weekday_is_saturday, weekday_is_sunday"

        print(insight1)
        print(insight2)
        print(insight3)
        print(insight4)
        print(insight5)
        print(insight6)

    @staticmethod
    def columns_numeric():
        cols_names_numeric = "n_tokens_content,num_hrefs,average_token_length,kw_avg_max,global_subjectivity,global_se
        return cols_names_numeric

    @staticmethod
```

Online news model

```
] : class OnlineShares:
    def __init__(self, filepath):
        self.filepath = filepath
        self.model_pipeline = Pipeline([
            ('scaler', StandardScaler()),
            ('regressor', RandomForestRegressor(n_estimators=160, max_depth=16, max_features='sqrt', min_samples_leaf=
        ])
        self.X_train, self.X_test, self.y_train, self.y_test = [None] * 4

    def load_data(self):
        self.data = pd.read_csv(self.filepath)
        print('Exploring dataset')
        DataExplorer.explore_data(self.data)
        print('Data wrangling tasks')
        DataExplorer.clear(self.data)
        print('Data visualization tasks')
        DataExplorer.basic_vis(self.data)
        print('Plotting histograms')
        DataExplorer.plot_histograms(self.data)
        print('Plotting correlation matrix')
        DataExplorer.plot_correlation_matrix(self.data)
        print('Other proposed plots')
        DataExplorer.otrosPlots(self.data)
        print('Transforming shares variable')
        DataExplorer.plots_after_transform(self.data)
        print('Applying tranformation to numerical features')
        DataExplorer.transformaciones_1(self.data)
        print('Applying tranformation to the last feature')
        DataExplorer.transformaciones_2(self.data)
        return self

    def preprocess_data(self):
        data = self.data
        X = data.drop('shares', axis=1)
        y = data['shares']
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

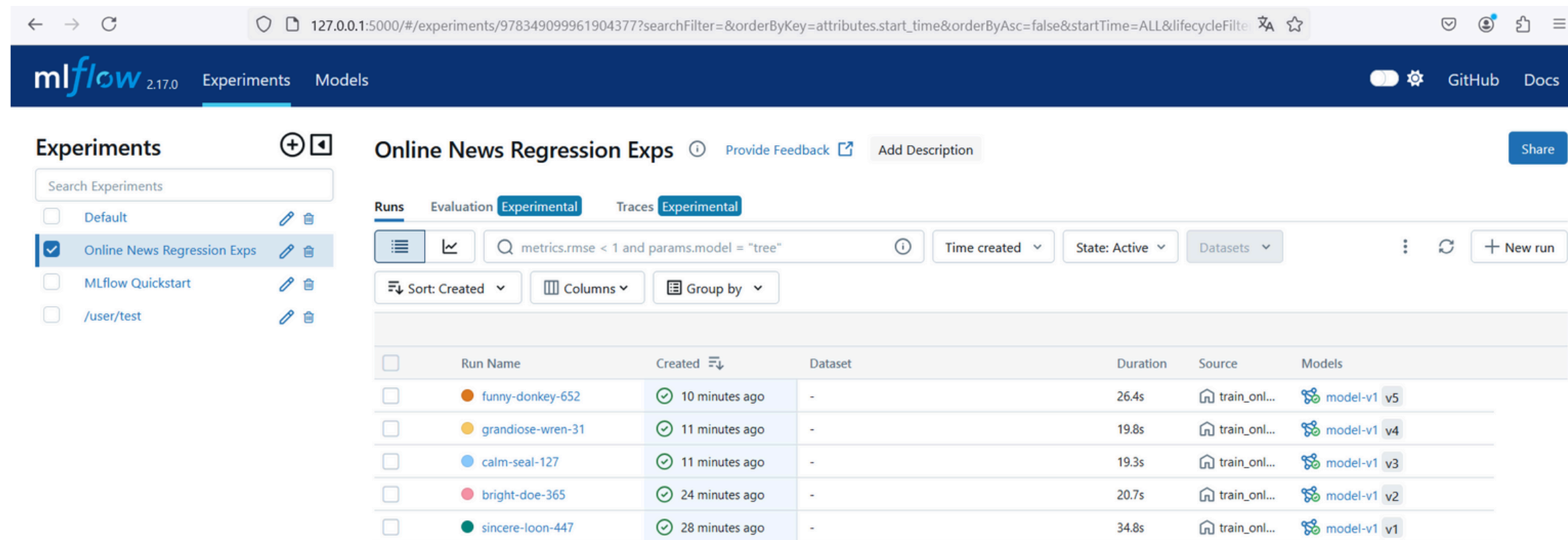
        # Transformaciones a columnas en X_train
        X_train['n_tokens_content'] = np.log1p(X_train['n_tokens_content'])
        X_train['num_hrefs'] = np.log1p(X_train['num_hrefs'])
        X_train['kw_avg_max'] = np.sqrt(X_train['kw_avg_max'])
        X_train['global_subjectivity'] = np.log1p(X_train['global_subjectivity'])
        X_train['global_rate_positive_words'] = np.sqrt(X_train['global_rate_positive_words'])

        # Columnas con Yeo-Johnson en X_train
        pt = PowerTransformer(method='yeo-johnson')

        X_train[['global_rate_negative_words', 'rate_positive_words', 'rate_negative_words']] = pt.fit_transform(
            X_train[['global_rate_negative_words', 'rate_positive_words', 'rate_negative_words']]
        )
        X_train['avg_positive_polarity'] = np.log1p(X_train['avg_positive_polarity'])
```

MLFLOW y Visualización de resultados

Para la parte de la visualización de resultados, decidimos utilizar la herramienta de Mlflow , ya que representa una alternativa tanto para el registro de modelo, métricas, hiperparámetros y demás cuestiones que son importantes para la hora de evaluar un modelo. Así como también su interfaz gráfica nativa con Python permite una apreciación en forma de dashboard del desempeño del modelo



The screenshot displays the MLflow web interface. The top navigation bar includes the MLflow logo (version 2.17.0) and tabs for 'Experiments' and 'Models'. The left sidebar shows a list of experiments, with 'Online News Regression Exps' selected. The main content area shows the details for this experiment, including a search bar, filters, and a table of runs.

Experiments

- ☐ Default
- ☒ Online News Regression Exps
- ☐ MLflow Quickstart
- ☐ /user/test

Online News Regression Exps

Runs | Evaluation | **Experimental** | Traces | **Experimental**

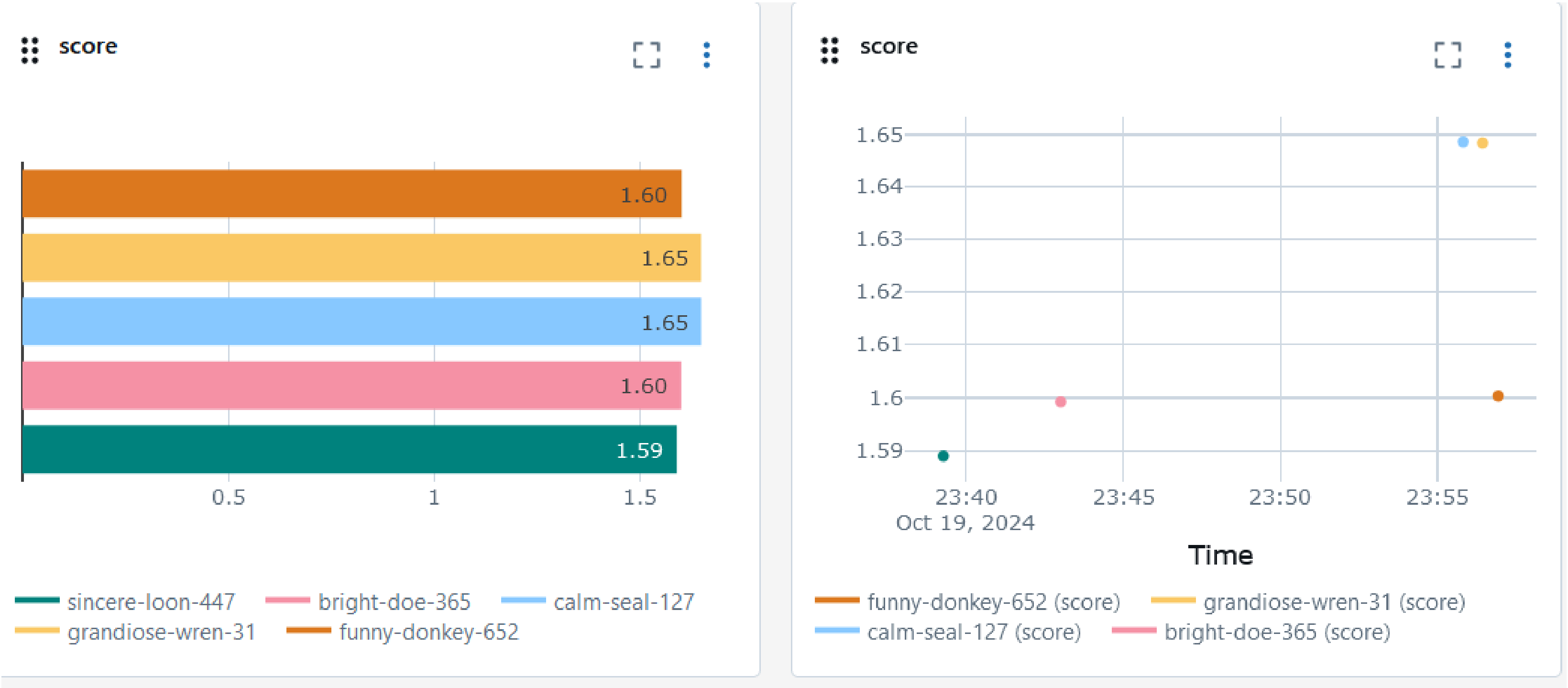
Search: metrics.rmse < 1 and params.model = "tree"

Sort: Created | Columns | Group by

Run Name	Created	Dataset	Duration	Source	Models
funny-donkey-652	10 minutes ago	-	26.4s	train_onl...	model-v1 v5
grandiose-wren-31	11 minutes ago	-	19.8s	train_onl...	model-v1 v4
calm-seal-127	11 minutes ago	-	19.3s	train_onl...	model-v1 v3
bright-doe-365	24 minutes ago	-	20.7s	train_onl...	model-v1 v2
sincere-loon-447	28 minutes ago	-	34.8s	train_onl...	model-v1 v1

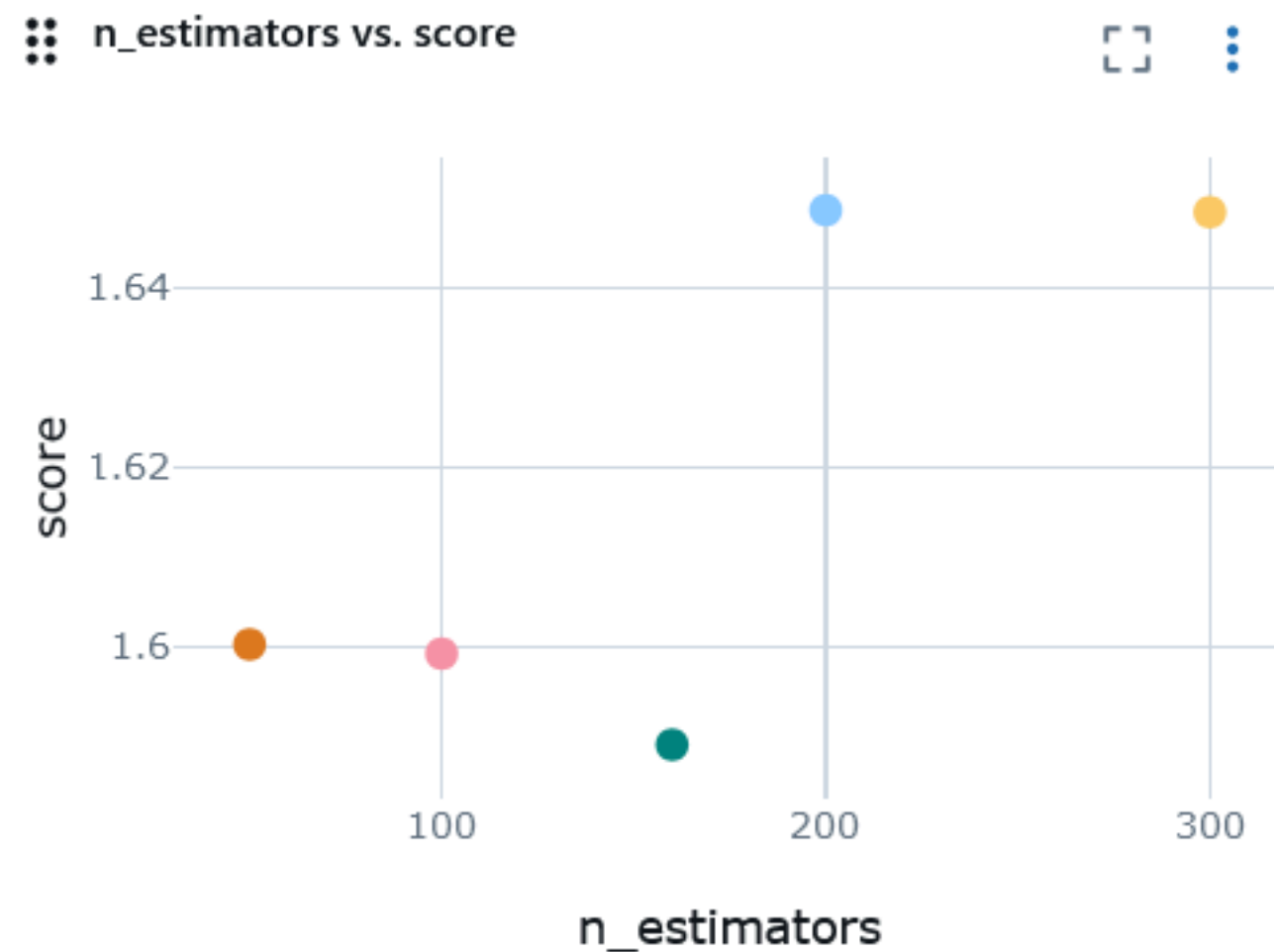
MLFLOW y Visualización de resultados

Mean Square Error

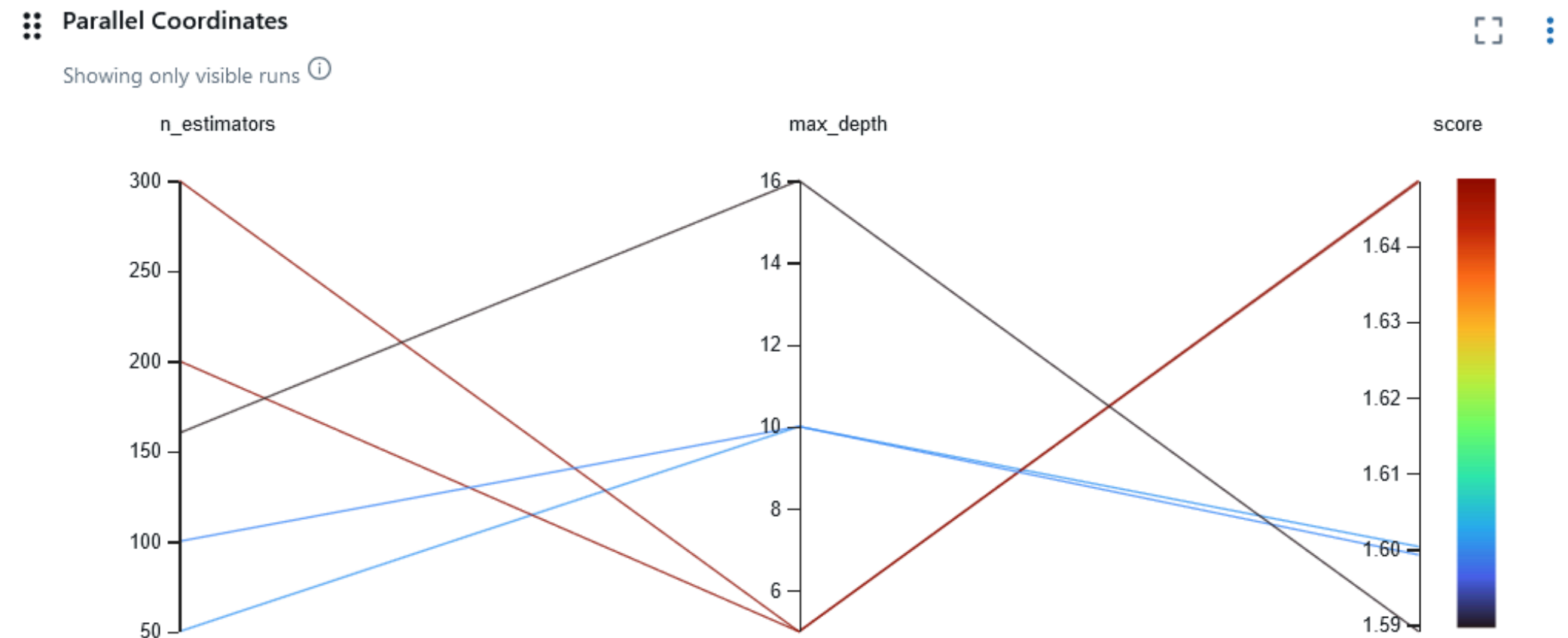


MLFLOW y Visualización de resultados

Con ayuda de la visualización podemos comparar también con los hiperparametros para darnos una idea de como hacer tuning al modelo



— sincere-loon-447 — bright-doe-365 — calm-seal-127
— grandiose-wren-31 — funny-donkey-652



MLOps Versionamiento

DVC

Para el versionamiento del dataset, habilitamos el uso de DVC el cual se configuro para utilizar AWS S3 como repositorio de información.

Politica IAM

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::test-dbt-01aws"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::test-dbt-01aws/*"
    }
  ]
}
```

Accesos

Configurar accesos

Les comparto por google drive estas credenciales.

```
dvc remote modify --local storage \
                    access_key_id 'mysecret'

dvc remote modify --local storage \
                    secret_access_key 'mysecret'
```

Configuración

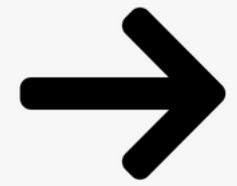
```
! config 1 x
.dvc > ! config > ...
1  [core]
2      remote = storage
3      autostage = false
4  ['remote "storage"']
5      url = s3://test-dbt-01aws/MLOPSGrupo3
6
```

Uso de espacio

```
PS C:\Users\jorge\OneDrive\Documents\GitHub\MLOPSGrupo3> dvc status
Data and pipelines are up to date.
PS C:\Users\jorge\OneDrive\Documents\GitHub\MLOPSGrupo3> dvc du .
6.00k .DS_Store
0      reports
0      a.py
0      docs
0      config
0      src
31.7M notebooks
1.24k .gitignore
0      models
1.15k README.md
68     .gitattributes
2      environment.yml
140    AUTHORS.md
31.7M
PS C:\Users\jorge\OneDrive\Documents\GitHub\MLOPSGrupo3>
```

```
online_news_popularity_clean.csv.dvc x
notebooks > data > processed > online_news_popularity_clean.csv.dvc
1  outs:
2  - md5: 1ef1ae54983ce1cc9d29cf70de907cb6
3    size: 8555629
4    hash: md5
5    path: online_news_popularity_clean.csv
6
```

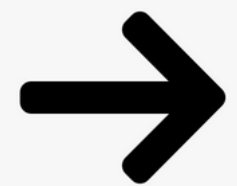
Conclusiones



Seguimiento de experimentos: mediante el uso de herramientas como DVC y MLflow, el seguimiento y el control de versiones de los experimentos de Machine Learning, se vuelven estructurados y eficientes. Esto garantiza que cada experimento esté bien documentado, lo que permite la reproducibilidad y la comparación entre diferentes ejecuciones.



Reproducibilidad mejorada: el control de versiones a través de herramientas como DVC ayuda a mantener conjuntos de datos, configuraciones y modelos. Esto permite una reversión y recreación más sencilla de experimentos anteriores, lo que garantiza resultados consistentes a lo largo del tiempo.



Información visual para tomar decisiones informadas: la visualización de métricas como la precisión y la exactitud proporciona información valiosa sobre el rendimiento de los modelos.



Registro de modelos organizado: al mantener un registro de modelos (MLflow), todos los modelos desarrollados se almacenan sistemáticamente con sus respectivas versiones y métricas de rendimiento. Esto permite una mejor gestión de los ciclos de vida de los modelos, lo que hace que estos sean más accesibles y fáciles de actualizar.