ITBA
Instituto Tecnológicc
de Buenos Aires

# SEGUNDO EXAMEN PARCIAL

# COMPETENCIA DE KAGGLE

Analítica Predictiva – 82.05

Paula González - 60784

# Limpieza de los datasets

- Fue realizada a través de **Rstudio**.
- **19 columnas** en el dataset **origen** y **18 columnas** en el dataset de **testeo**.
- Para las variables categóricas realicé **frequency encoding** para pasarlos a datos numéricos.
- Los **NAs** fueron **imputados** con las **medias** de las variables numéricas.
- Los datos de runtimeMinutes en cero fueron reemplazados por la media de la variable.

| | averageRating | numVotes | isAdult | startYear | endYear | runtimeMinutes |
|---|---|---|---|---|---|---|
| 1 | 4.4 | 15 | 0 | 1951 | 0 | 91.00000 |
| 2 | 7.0 | 990 | 0 | 2007 | 2021 | 30.00000 |
| 3 | 8.1 | 41 | 0 | 2011 | 0 | 44.00000 |
| 4 | 4.6 | 48 | 0 | 1969 | 0 | 84.00000 |
| 5 | 5.6 | 28 | 0 | 2010 | 0 | 130.00000 |
| 6 | 9.3 | 43 | 0 | 2013 | 0 | 19.00000 |
| 7 | 7.1 | 10 | 0 | 2016 | 0 | 45.00000 |

| cantidad_directors | frec_directors | frec_titleType | frec_genres_x | cantidad_generos |
|---|---|---|---|---|
| 1 | 2.966627e-05 | 0.241080425 | 1.411705e-03 | 2 |
| 4 | 2.045950e-06 | 0.066576236 | 2.828526e-02 | 3 |
| 1 | 1.534462e-05 | 0.448325953 | 1.074124e-03 | 3 |
| 1 | 2.045950e-06 | 0.241080425 | 7.342710e-02 | 1 |
| 1 | 1.432165e-05 | 0.241080425 | 1.902017e-02 | 2 |
| 1 | 5.114875e-06 | 0.114487270 | 2.045950e-05 | 3 |
| 1 | 1.505185e-01 | 0.448325953 | 4.917441e-03 | 3 |
| 1 | 1.196881e-04 | 0.448325953 | 6.332215e-04 | 3 |

# Limpieza de los datasets

Quedaron los siguientes **features**:

| Originales | Nuevos |
|---|---|
| numVotes | cantidad_writers |
| isAdult | cantidad_directors |
| startYear | frec_directors |
| endYear | frec_titleType |
| runtimeMinutes | frec_genres_x |
| seasonNumber | cantidad_géneros |
| episodeNumber | frec_ori_language |
| popularity | frec_companies |
| | duration |

# Modelo Baseline

- Elegí el modelo de regresión lineal para usar como baseline.
- Tuvo un score de 0.08

```
import os
import pandas as pd
import numpy as np
from sklearn import linear_model
```
✓ 9.1s

```
#Creo un modelo regresion lineal
modelo_regresion = linear_model.LinearRegression()
# Entrenar el modelo
modelo_regresion.fit(X,y)
# Realiza las predicciones del conjunto de entrenamiento
Yest = modelo_regresion.predict(X)
```

```
#Saco el score
modelo_regresion.score(X,y)
```
[8] ✓ 0.2s

```
0.08411775230387053
```

# Prueba de XGBOOST

1. Empecé probando un modelo XGBOOST sin hiperparámetros.
2. Me dio un score de -0.011 que estaba asociado al frecuency encoding que realicé sobre las variables categóricas.
3. Una vez corregido eso, obtuve un score de 0.33.
4. Fui agregando hiperparámetros y probando distintos valores en ellos. De todas las pruebas, el mayor score que me lanzó fue un 0.38.

```
regresion = XGBRegressor()

regresion.fit(X_train, y_train)
score = regresion.score(X_test, y_test)

print("R^2 score on testing data: {:.4f}".format(score))
```

✓ 2.5s

```
R^2 score on testing data: 0.3331
```

```
regresion = XGBRegressor(
    n_estimators=100,  # Número de árboles de decisión
    learning_rate=0.5,  # Tasa de aprendizaje
    max_depth=12,  # Profundidad máxima del árbol
    subsample=0.8,  # Muestra de filas
    colsample_bytree=0.8  # Muestra de columnas
)

# Entrenas el modelo y evalúas el rendimiento
regresion.fit(X_train, y_train)
score = regresion.score(X_test, y_test)

print("R^2 score on testing data: {:.4f}".format(score))
```

✓ 7.0s

```
R^2 score on testing data: 0.3823
```

# Modelo final con RandomForest

1. Empecé probando un modelo RandomForest sin hiperparámetros.

2. Me dio un score de 0.42

3. Fui agregando hiperparámetros y probando distintos valores en ellos. De todas las pruebas, el mayor score que me lanzó fue un 0.43.

```
regresion = RandomForestRegressor()

regresion.fit(X_train, y_train)
score = regresion.score(X_test, y_test)

print("R^2 score on testing data: {:.4f}".format(score))
```

✓ 8m 19.9s

```
R^2 score on testing data: 0.4237
```

```
hiperparametros = {
    'n_estimators': 100,  # Número de árboles en el bosque
    'max_depth': None,
    # Profundidad máxima de los árboles (None implica que los árboles se expandirán hasta que todas las hojas contengan menos de min_samples_split muestras)
    'min_samples_split': 3,  # Número mínimo de muestras requeridas para dividir un nodo interno
    'min_samples_leaf': 1,   # Número mínimo de muestras requeridas para estar en un nodo hoja
    'max_features': 'sqrt',  # Cambiado de 'auto' a 'sqrt' (puedes usar también 'auto' o algún número entero)
    'random_state': 42  # Semilla para reproducibilidad
}

regresion = RandomForestRegressor(**hiperparametros)

regresion.fit(X_train, y_train)
score = regresion.score(X_test, y_test)

print("R^2 score on testing data: {:.4f}".format(score))
```

✓ 2m 47.1s Pyth

```
R^2 score on testing data: 0.4325
```

# Limitaciones y posibles mejoras

- Tardé mucho tiempo en limpiar la base intentando de trabajar con los datos en formato json.
- La próxima vez intentaría limpiar la base de datos directamente en el entorno de Python, para ahorrar tiempo en el ida y vuelta entre los programas y el tiempo de la subida de datos en VisualStudio Code.
- Dediqué dos días completos a la configuración de un código para la búsqueda de hiperparámetros óptimos que se adapten a mi modelo. Sin embargo, el proceso no se completó satisfactoriamente y quedó inconcluso.
- En caso de disponer de tiempo adicional, habría profundizado en la investigación sobre la aplicación de un modelo Support Vector Machine.

ITBA
Instituto Tecnológicc
de Buenos Aires

# ¡MUCHAS GRACIAS!

MÁS INFORMACIÓN > **www.itba.edu.ar**